

INFORMATION ABOUT ORPHANAGES

AN INDUSTRIAL INTERNSHIP REPORT

Submitted in partial fulfilment for the award of the degree of

MTech

in

Software Engineering

by

PANDIRI JOHN PAUL (16MIS0375)



VIT[®]

Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

School of Information Technology and Engineering

Department of Software and Systems Engineering

MAY 2020



VIT[®]
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

School of Information Technology and Engineering
Department of Software and Systems Engineering

DECLARATION BY THE CANDIDATE

I hereby declare that the Industrial Internship report entitled **“INFORMATION ABOUT ORPHANAGES”** submitted by me to VIT, Vellore, in partial fulfillment of the requirement for the award of the degree of **M Tech (Software Engineering)** is a record of bonafide **Industrial Internship -SWE3099** carried out by me under the guidance of **Vasanthi BDE**. I further declare that the work reported in this project has not been submitted and will not be submitted, either in part or in full, for the award of any other degree in this institute or any other institute or university.

Place: Vellore

Date: 11th May 2020

Signature of the Candidate



VIT[®]
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

School of Information Technology and Engineering
Department of Software and Systems Engineering

BONAFIDE CERTIFICATE

This is to certify that the Industrial Internship report entitled “**INFORMATION ABOUT ORPHANAGES**” by **PANDIRI JOHN PAUL (16MIS0375)** to VIT, Vellore, in partial fulfillment of the requirement for the award of the degree of **M Tech (Software Engineering)** is a record of bonafide work carried out by him /her under my guidance. The project fulfills the requirements as per the regulations of this Institute and in my opinion meets the necessary standards for submission. The contents of this report have not been submitted and will not be submitted either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university.

Signature of Internal Guide
Prof: Santhosh Kumar SVN

Examiner(s) Signature

- 1.**
- 2.**



Abstract

Analysis of providing information about orphanages is a computer based system used to provide the information about the orphanages based on donations sent to the orphanages. System is mainly for the systematic usage of the website by the user and the admin. The objective of this application is to develop a centralized website for orphanages. To provide facilities of various social activities in a single website. People can donate through online.

It also provides accurate and regular information about the orphanage. This helps us to maintain a record of all the orphanages which can be used for sending donations. It also includes banking details of both the sender and the orphanage. In the existing system the donations include lot of paper work and man work. But the proposed system eliminates all the manual work done by the donors. The donors no need to visit the orphanages for donation they can directly donate through the system. For donating an item we need to visit the orphanage by finding the orphanage address and deliver the item. But in the proposed system we can deliver the item directly through the vendor. We need to place an order and enter the address of the orphanage so the item directly delivered to the orphanage. The proposed system increases the effectiveness and efficiency by reducing the work intensity. It also increases the accuracy of computation and record maintenance. By maintain a centralized database we can increase the optimization and reduce the use of paper work.

We are providing a platform for the donors and the orphanages to check the donation details and information about the orphanage like address and other bank details for transaction. We need to make a secure web application so that the bank transaction details are not seen by unauthorized users. Ease of update and maintenance of operation. We can easily update all the details in database if we are an authorized user.

ACKNOWLEDGEMENT

I wish to express our heartfelt gratitude to **Dr.G.Viswanathan**, Chancellor, VIT, Vellore, for providing facilities for the Industrial Internship. I am highly grateful to our Vice President, **Dr.G. Sekar Viswanathan**, Vice chancellor **Dr. Anand A. Samuel**, and Pro-Vice Chancellor **Dr.S.Narayanan**, for providing the necessary resources.

My sincere gratitude to **Dr. Balakrushna Tripathy**, Dean, School of Information Technology and Engineering, for giving me the opportunity to undertake the project.

I wish to express my sincere gratitude to **Dr. S. Sree Dharinya**, Head of the Department, Software and Systems Engineering, **Prof. Elango N M & Prof. Sujata R**, Industrial Internship Coordinators, M.Tech (Software Engineering), School of Information Technology and Engineering for providing me continuous support to do my project work.

I would like to express my special gratitude and thanks to my external guide **Mrs Vasanthi, BDE, Grepthor software solutions** and internal guide **Prof. Santhosh Kumar SVN, Assistant Professor, SITE** for their esteemed guidance, immense support and encouragement to complete the internship successfully.

I thank the management of VIT, Vellore for permitting me to use the library resources. I also thank all the faculty members of VIT, Vellore for giving me the courage and strength I needed to complete my goals. This acknowledgement would be incomplete without expressing my whole hearted thanks to my family and friends who motivated me during the course of the work.

Place: Vellore

Date: 11th May 2020

PANDIRI JOHN PAUL

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	ABSTRACT	
	LIST OF TABLES	ix
	LIST OF FIGURES	x
	LIST OF SYMBOLS	xi
1.	INTRODUCTION	
	1.1.Problem Statement	1
	1.2.Motivation	1
	1.3.Objective	1
	1.3.1.Proposed System	2
	1.3.2.Advantages of Proposed system	2
2.	TECHNOLOGIES LEARNT	3
3.	SYSTEM DESIGN	
	3.1 System Architecture	8
	3.2 Module description	9
	3.3 System Specification	10
	3.3.1 Software Requirements	10
	3..3.2 Hardware Requirements	10
	3.4 Detailed Design	
	3.4.1 Use case Diagram	11
	3.4.2 Sequence Diagram	12
	3.4.3 Class Diagram	13
	3.4.4 Dataflow diagram	13

	3.4.5 Activity diagram	14
4.	IMPLEMENTATION	15
	4.1 Implementation details	
5.	TEST RESULTS	18
	5.1 Test cases	
6.	RESULTS AND DISCUSSIONS	19
7.	CONCLUSION AND FUTURE WORK	23
	7.1 Conclusion	
	7.2 Future Work	
8.	REFERENCES	23

LIST OF TABLES

Table No	Title	Page No
1.1	Test cases	18

LIST OF FIGURES

Figure No	Title	Page No
1	System Architecture	8
2	Use case	11
3	Class diagram	13
4	Sequence diagram	12
5	Data flow diagram	13
6	Activity diagram	14

LIST OF ABBREVIATIONS

ACRONYM	EXPANSION
MVT	Models View Templates
HTML	Hyper Text Markup Language
XML	Extensible Markup language
HTTP	Hyper Text Transfer Protocol
URL	Uniform Resource Locator
CSS	Cascading Style Sheets
SQL	Structured Query Language

1. Introduction

Computerization is the activity of facilitating or automating procedures or activities by means of electronic computer. When we talk about a computerized system, we are referring to a function, process or operation integrated with a computer system and performed by trained people. The world is moving at an unimaginable speed in the area of information use and dissemination. So we need to provide a system which helps to ease the maintenance of details of orphanages for donation purpose.

1.1 Problem Statement:

The problems of existing system are numerous but a few issues are like maintaining the records of the donations manually and filling the details is fraught with delay. Since the existing system consists of information return in forms they can be bound to duplication of information which leads to inaccuracy. Lack of database in existing system leads to losing of important documents and information regarding payments and other details. Inefficiency in records maintaining leads to manual filling of all the details regarding details of orphanages.

1.2 Motivation

Purpose of the project is to provide a system which helps to ease the maintenance of details of orphanages. It should also maintain the record of the donations, donors and items donated. It should remove the inefficiency present in the existing system. So we need to create a web based application which will provide all the features that improves the effectiveness of the system. A centralized database helps us to improve the data efficiency.

1.3 Objective

We need to provide a platform for the donors and the orphanages to check the donation details and information about the orphanage like address and other bank details for transaction. We need to make a secure web application so that the bank

transaction details are not seen by unauthorized users. We need to create a user friendly application so that all the users can easily use it. This application helps the user to donate directly to the orphanage by using the bank details of orphanages. The main objective of this project is to provide a reliable solution to the problems of manual record system and manual transaction of money to the orphanages.

1.3.1 Proposed System

The proposed system consists of phases like donor, receiver and vendor. The whole process is done between these phases. Each phase needs a record system for maintaining the details involved during the process done. The roles of each phase are

Donor: This person needs to give the details of his/her for item donation or money donation to the orphanage. They need to select the orphanage from the form given in the system and continue with their process.

Receiver: This phase includes the details of the orphanage and the location. Depending on the type of donation the orphanage should provide the details for further transaction. These details are present in the database. When we select the mode of transaction then we can proceed with further step.

Vendor: The donor can also donate items by contacting the vendor and deliver the items to the orphanages. For this process the vendor need to collect the details of orphanage from the donor. This requires database so that the donor can directly add the address of orphanage in the web application.

So by using the web application we can skip all the manual work and efforts.

1.3.2 Advantages of proposed system

Effectiveness and efficiency by reducing the work intensity. In the proposed system we have a centralized database in which we can store all the data directly which resolves the issues of paper work and loss of data.

Accuracy of computation and record keeping. While sending items or money we have few issues like address mistakes or bank details. By using a web based application and database we can save the details directly in a database and use them wherever required.

Centralized database. By maintaining a database many problems can be resolved as mentioned above.

Speed optimization and reduced usage of paper work. The details of donor, receiver and the vendor can be stored in database and the transaction details are stored directly in database which reduces lots of paper work and physical work.

Ease of update and maintenance of operation. We can easily update all the details in database if we are an authorized user.

2. Technologies learnt:

Technologies learnt while doing this project are:

- Django
- Python
- Html Forms implementing in Django framework.
- Database connection.

Django is a Python-based free and open-source web framework, which follows the model-template-view architectural pattern. It is maintained by the Django Software Foundation, an independent organization established as a non-profit.

Django's primary goal is to ease the creation of complex, database-driven websites. The framework emphasizes reusability and plugability of components, less code, low coupling, rapid development, and the principle of don't repeat yourself. Python is used throughout, even for settings files and data models. Django also provides an

optional administrative create, read, update and delete interface that is generated dynamically through introspection and configured via admin models.

Because Django was developed in a fast-paced newsroom environment, it was designed to make common Web-development tasks fast and easy. Django provides an abstraction layer the “models” for structuring and manipulating the data of your Web application.

❖ Model:

A model is the single, definitive source of information about your data. It contains the essential fields and behaviours of the data you’re storing. Generally, each model maps to a single database table.

The basics:

- Each model is a Python class that sub classes `django.db.models.Model`.
- Each attribute of the model represents a database field.
- With all of this, Django gives you an automatically-generated database-access API

❖ Views

Django has the concept of “views” to encapsulate the logic responsible for processing a user’s request and for returning the response.

A view function, or view for short, is simply a Python function that takes a Web request and returns a Web response. This response can be the HTML contents of a Web page, or a redirect, or a 404 error, or an XML document, or an image . . . or anything, really. The view itself contains whatever arbitrary logic is necessary to return that response. This code can live anywhere you want, as long as it’s on your Python path. To design URLs for an app, you create a Python module informally

called a URLconf (URL configuration). This module is pure Python code and is a mapping between URL path expressions to Python functions (your views).

This mapping can be as short or as long as needed. It can reference other mappings. And, because it's pure Python code, it can be constructed dynamically.

❖ Template:

Being a web framework, Django needs a convenient way to generate HTML dynamically. The most common approach relies on templates. A template contains the static parts of the desired HTML output as well as some special syntax describing how dynamic content will be inserted. The Django template language is Django's own template system. Until Django 1.8 it was the only built-in option available. It's a good template library even though it's fairly opinionated and sports a few idiosyncrasies.

Django's template language is designed to strike a balance between power and ease. It's designed to feel comfortable to those used to working with HTML. If you have any exposure to other text-based template languages, such as Smarty or Jinja2, you should feel right at home with Django's templates.

❖ HTML forms

In HTML, a form is a collection of elements inside `<form>...</form>` that allow a visitor to do things like enter text, select options, manipulate objects or controls, and so on, and then send that information back to the server.

Some of these form interface elements - text input or checkboxes - are fairly simple and are built into HTML itself. Others are much more complex; an interface that pops up a date picker or allows you to move a slider or manipulate controls will typically use JavaScript and CSS as well as HTML form `<input>` elements to achieve these effects.

The login form for the Django admin contains several `<input>` elements: one of `type="text"` for the username, one of `type="password"` for the password, and one of `type="submit"` for the “Log in” button. It also contains some hidden text fields that the user doesn’t see.

GET and POST are the only HTTP methods to use when dealing with forms.

Django’s login form is returned using the POST method, in which the browser bundles up the form data, encodes it for transmission, sends it to the server, and then receives back its response.

GET, by contrast, bundles the submitted data into a string, and uses this to compose a URL. The URL contains the address where the data must be sent, as well as the data keys and values.

❖ Projects and Applications in Django:

The term project describes a Django web application. The project Python package is defined primarily by a settings module, but it usually contains other things. For example, when you run `django-admin startproject mysite` you’ll get a mysite project directory that contains a mysite Python package with `settings.py`, `urls.py`, and `wsgi.py`. The project package is often extended to include things like fixtures, CSS, and templates which aren’t tied to a particular application.

❖ Python:

Python is a multi-paradigm programming language. Object-oriented programming and structured programming are fully supported, and many of its features support functional programming and aspect-oriented programming including by metaprogramming and metaobjects. Many other paradigms are supported via extensions, including design by contract and logic programming.

Python uses dynamic typing, and a combination of reference counting and a cycle-detecting garbage collector for memory management. It also features dynamic name resolution, which binds method and variable names during program execution.

❖ Database:

Setting up a database in django framework.

Open up `mysite/settings.py`. It's a normal Python module with module-level variables representing Django settings.

By default, the configuration uses SQLite. If you're new to databases, or you're just interested in trying Django, this is the easiest choice. SQLite is included in Python, so you won't need to install anything else to support your database. When starting your first real project, however, you may want to use a more scalable database like PostgreSQL, to avoid database-switching headaches down the road.

If you wish to use another database, install the appropriate database bindings and change the following keys in the DATABASES 'default' item to match your database connection settings:

ENGINE – Either `'django.db.backends.sqlite3'`, `'django.db.backends.postgresql'`, `'django.db.backends.mysql'`, or `'django.db.backends.oracle'`.

❖ MYSQL:

MySQL is an open-source relational database management system (RDBMS). Its name is a combination of "My", the name of co-founder Michael Widenius's daughter, and "SQL", the abbreviation for Structured Query Language.

MySQL is free and open-source software under the terms of the GNU General Public License, and is also available under a variety of proprietary licenses. MySQL was owned and sponsored by the Swedish company MySQL AB, which was bought by Sun Microsystems. In 2010, when Oracle acquired Sun, Widenius forked the open-source MySQL project to create MariaDB.

MySQL is a component of the LAMP web application software stack, which is an acronym for Linux, Apache, MySQL, Perl/PHP/Python. MySQL is used by many database-driven web applications, including Drupal, Joomla, phpBB, and WordPress.

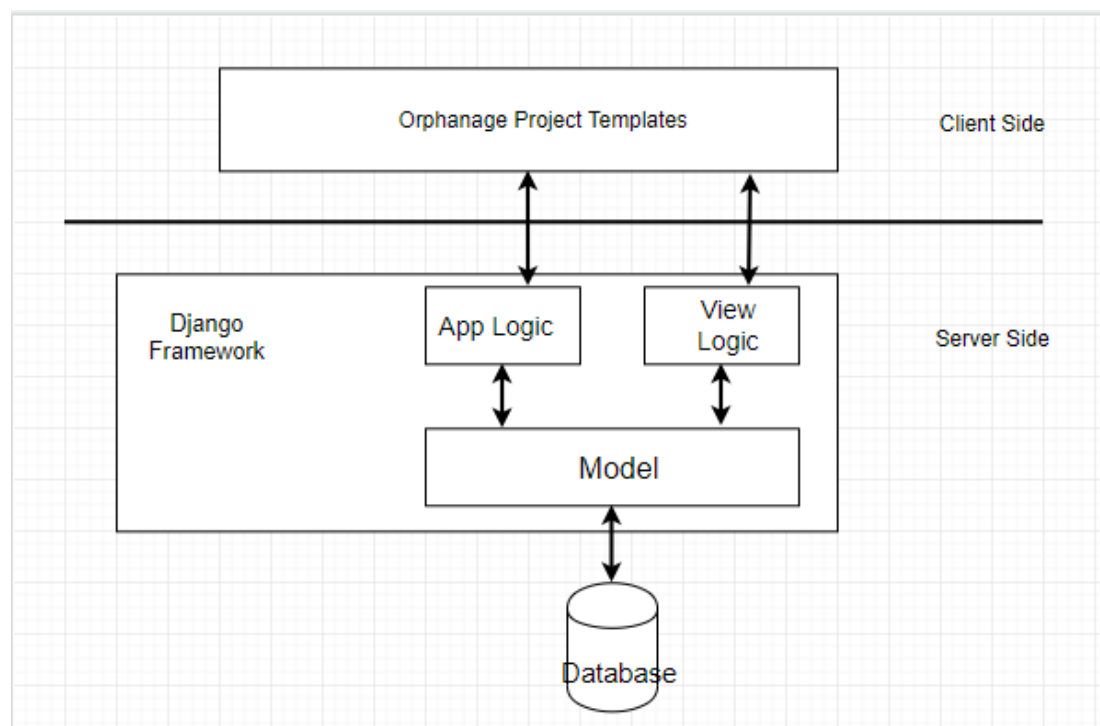
MySQL is written in C and C++. Its SQL parser is written in yacc, but it uses a home-brewed lexical analyzer. MySQL works on many system platforms, including AIX, BSDi, FreeBSD, HP-UX, eComStation, i5/OS, IRIX, Linux, macOS, Microsoft Windows, NetBSD, Novell NetWare, OpenBSD, OpenSolaris, OS/2 Warp, QNX, Oracle Solaris, Symbian, SunOS, SCO OpenServer, SCO UnixWare, Sanos and Tru64. A port of MySQL to OpenVMS also exists.

The MySQL server software itself and the client libraries use dual-licensing distribution. They are offered under GPL version 2, or a proprietary license.

Oracle offers paid support via its MySQL Enterprise products. They differ in the scope of services and in price. Additionally, a number of third party organisations exist to provide support and services, including MariaDB and Percona. When using some storage engines other than the default of InnoDB, MySQL does not comply with the full SQL standard for some of the implemented functionality, including foreign key references.

3. System Design

3.1 System Architecture



3.2 Module Description

Home Module: This module consists of login and registration module. We can use these two to get access to the web application. We need to register first and create a user id to use the login module to access into application. We will get login details when we register by entering our details like email, contact and date of birth.

Login Module: Here the donor need to login into the web application to donate items or money to the orphanages. The user need a username and password to log in to the application. The remaining modules included in the application can be viewed after this module. This module prevents the unauthorized users to access the data present in the database.

Registration Module: New users can register their details in registration module. The new users should set their new username and password. Few other details like email, contact and date of birth are entered for creating a new user id. After submitting the mentioned details then a new user id is created which can be used for login. All the details entered in this module are saved in the database. For any queries about the user we can view them in the database.

Items Donation module: The user or donor can log into the web based application and donate money or items. If the donor want to donate any items he/she can choose items donation and choose the item type that they want to donate. The donor should enter the shop address and contact where the item is available. Address of orphanage should also be entered so that the item is delivered to correct location. After submitting the details of item location and delivery location then the payment process is done. Then the donation is completed.

Payment module: If the donor wants to donate money to the orphanage then they need to submit details like orphanage name and the amount they want to donate. The donor should also submit few personal details like donor name, donor email and donor contact. They can also select the type of payment and continue the transaction. This module is linked with the pay module which consists of money transaction.

Pay Module: In this module the money transaction takes place. The donor should enter the details of bank account like card number, card holder name, expiry date and cvv. After this the page redirects to secure page where the transaction takes place.

3.3 System Specification

All system developed has a predefined system requirement for a maximum performance. However, the system requirement is the minimum hardware and software to be designed. Below are the minimum hardware and software for the development.

3.3.1 Software Requirements

This project requires a software requirements like

- i. Minimum of windows operating system.
- ii. MySQL database.
- iii. Internet services.
- iv. Django framework.
- v. Python 2.7 in any IDE like Pycharm or eclipse.

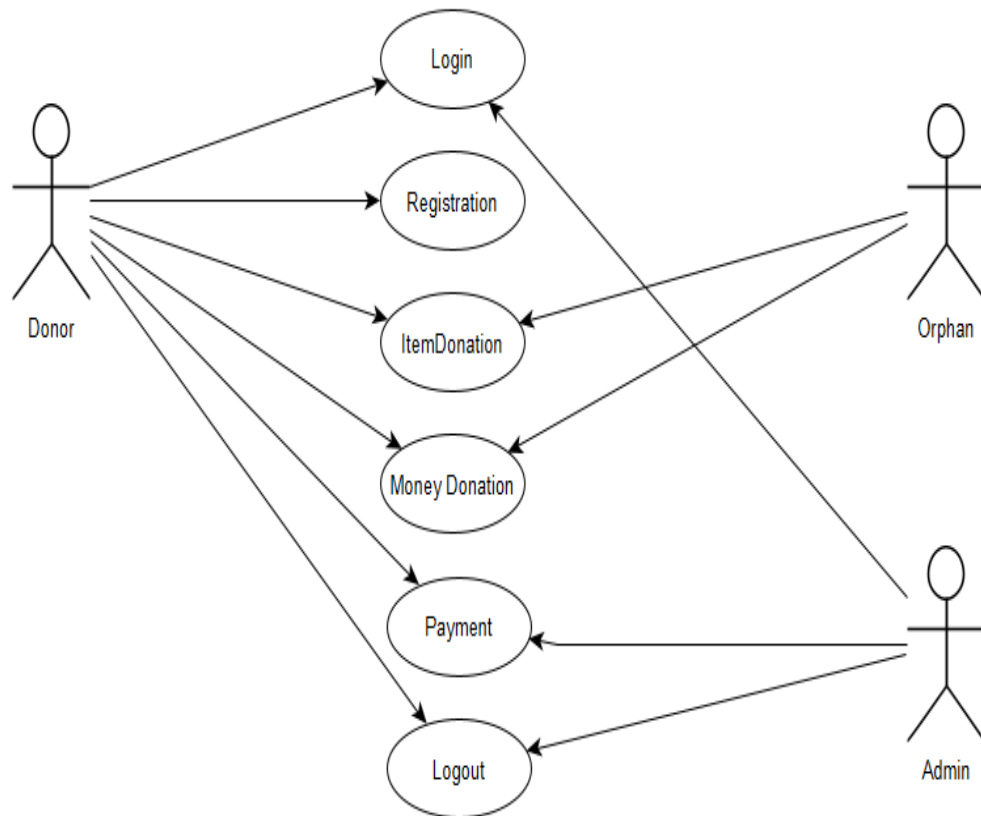
3.3.2 Hardware Requirements

This project requires hardware requirements like

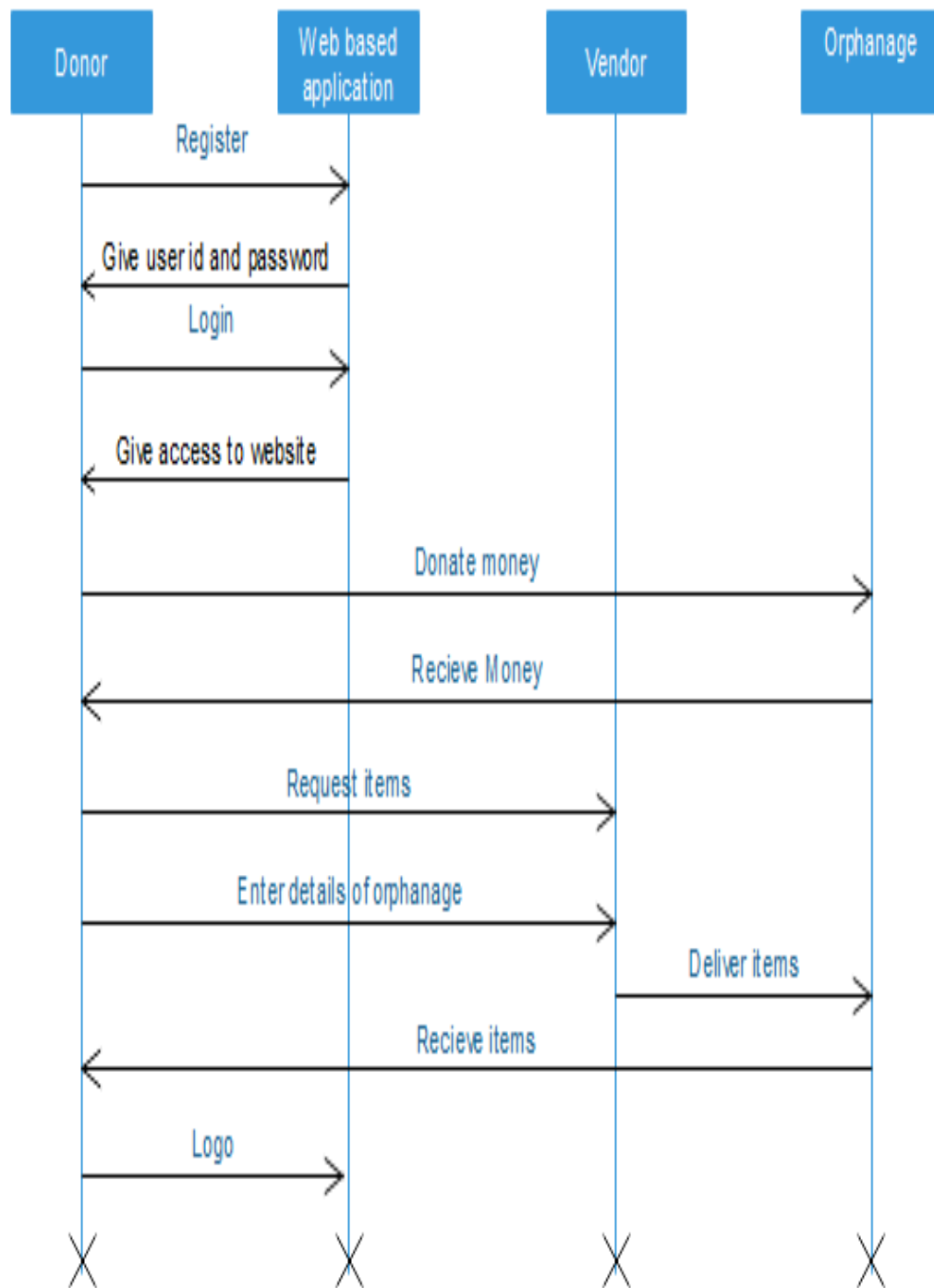
- i. A personal computer.
- ii. Minimum RAM of 2GB.
- iii. Uninterrupted power supply.
- iv. Processor of minimum 1GHz.
- v. Hard Drive of at least 64GB.
- vi. Wireless adapter or Ethernet connection.

3.4 Detailed Design

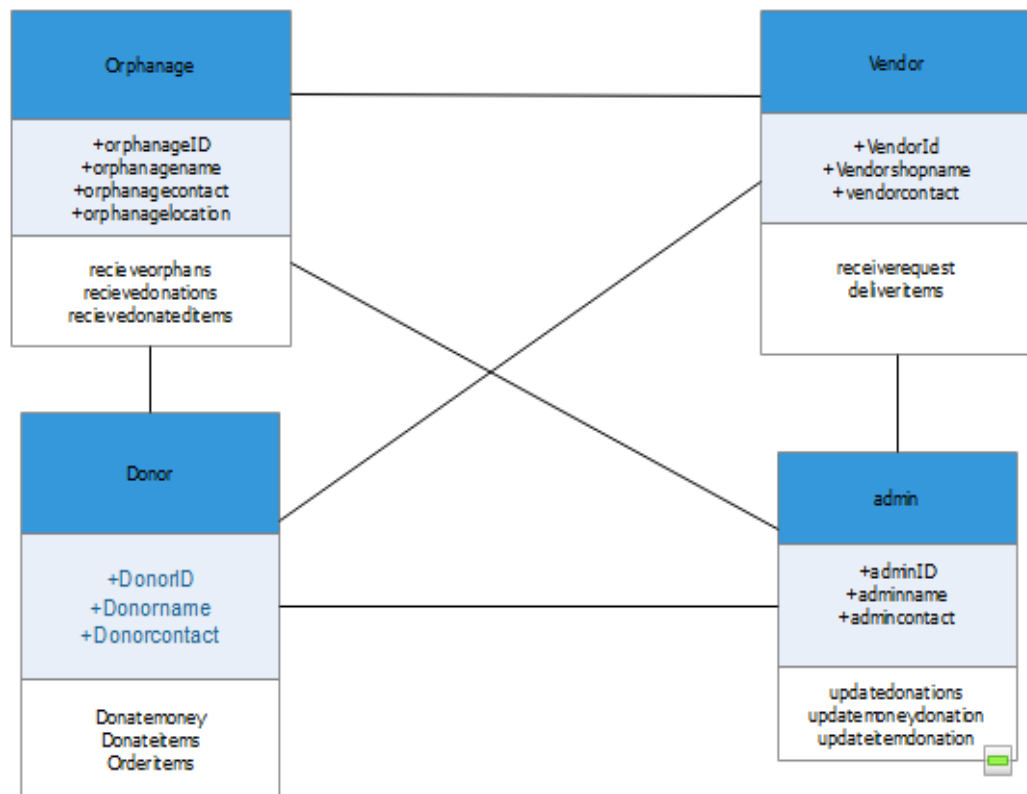
3.4.1 Use case diagram



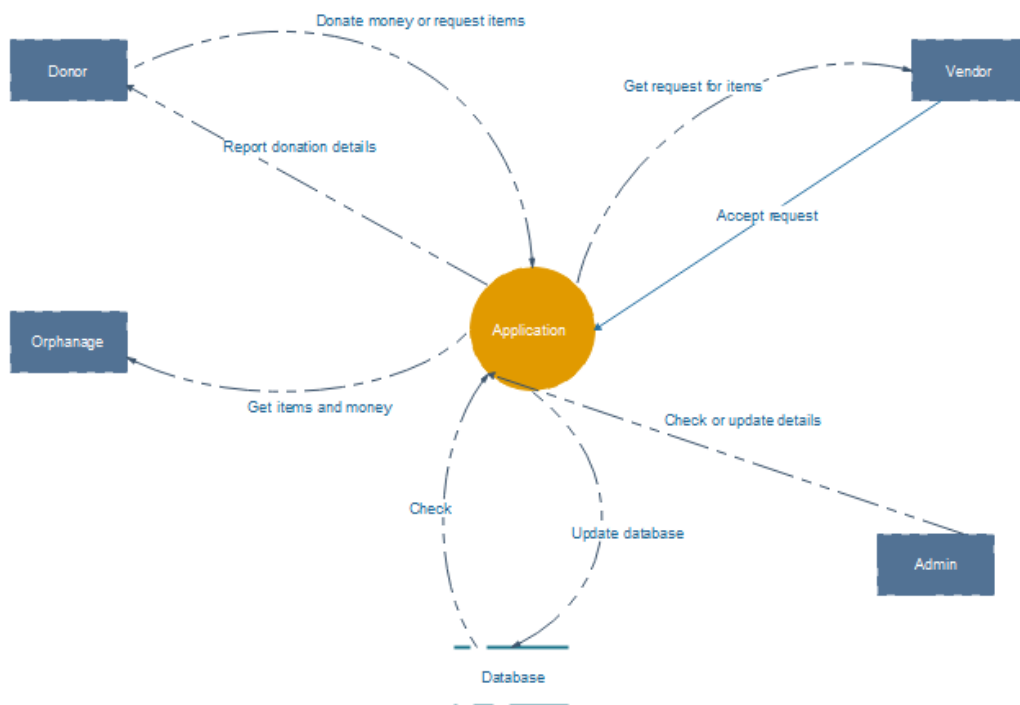
3.4.2 Sequence Diagram



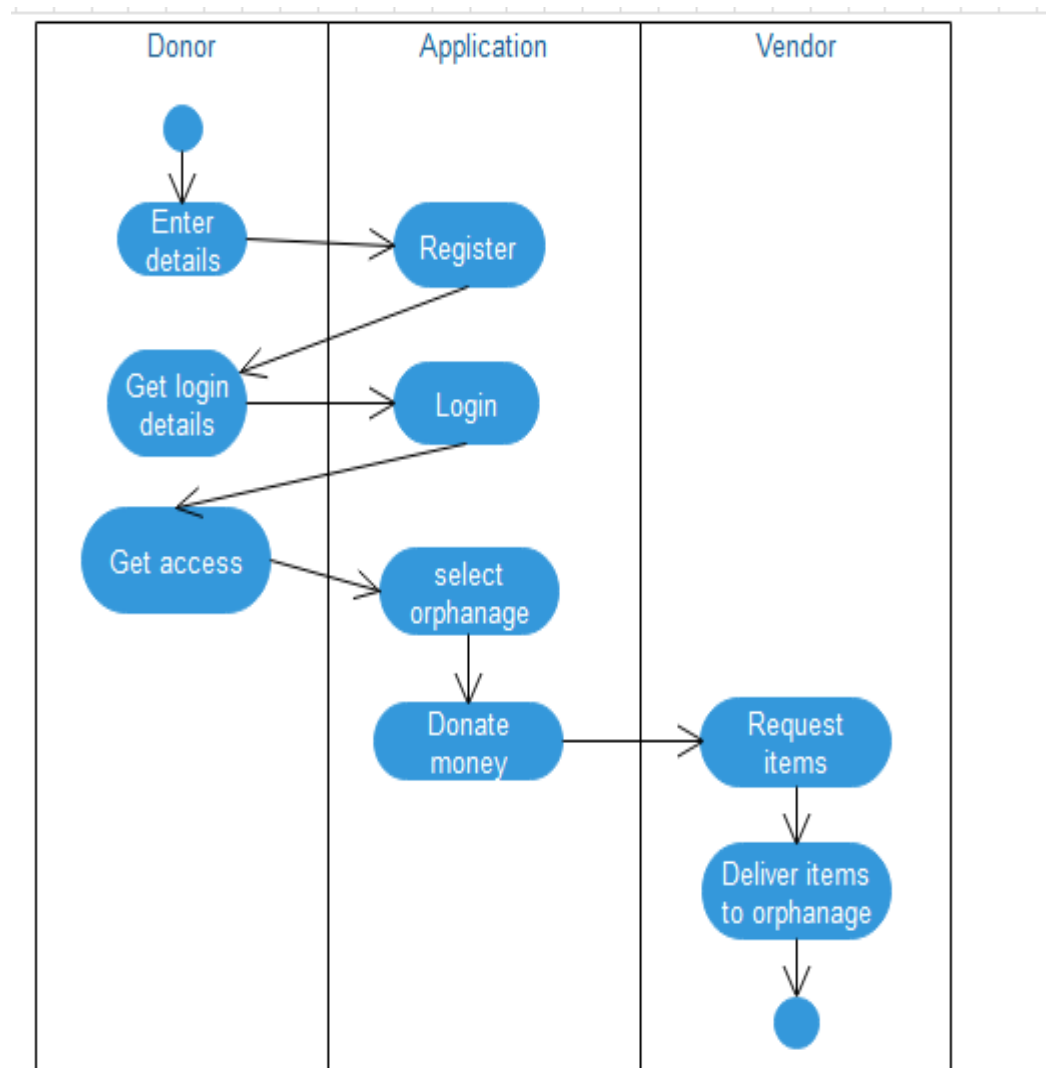
3.4.3 Class Diagram



3.4.4 Dataflow diagram



3.4.5 Activity Diagram



4. Implementation

The project is implemented in Django framework using python, few html forms and a database.

4.1 Implementation details

This project was implemented in Python using Pycharm or visual studio IDE. The project is implemented in following way. It consists of five types of webpages like Login, registration, Item donation, Payment and Pay. This project help the donor to donate to the orphanage directly through this wen application without any third party interference. The details of the orphanage are stored in the database.

4.2 Pseudo Code:

forms.py:

```
from django import forms
from .models import Registration
from .models import itemsdonation
from .models import Payment

States= [
    ('States', 'States'),
    ('Andhra Pradesh', 'Andhra Pradesh'),
    ('Telangana', 'Telangana'),
    ('Arunachal Pradesh', 'Arunachal Pradesh'),
    ('Kerala', 'Kerala'),

class RegistrationForm(forms.ModelForm):

    class Meta:
        model = Registration
        fields = '__all__'

class LoginForm(forms.Form):
    username=forms.CharField(max_length=30)
    password=forms.CharField(max_length=30)

class itemsdonationForm(forms.ModelForm):
    class Meta:
        model=itemsdonation
        fields='__all__'
    States=forms.CharField(label='Select the States',
                           widget=forms.Select(choices=States))
```

```

        Areas=forms.CharField(label='Select the Areas',
                               widget=forms.Select(choices=Areas))
class PaymentForm(forms.ModelForm):
    class Meta:
        model=Payment
        fields='__all__'
    Payments= forms.CharField(label='Select the Payments',
                              widget=forms.Select(choices=Payments))

class PayForm(forms.Form):
    cardholder=forms.CharField(max_length=30)
    cardnum=forms.CharField(max_length=30)
    cvv=forms.CharField(max_length=3)
    expdate=forms.CharField(max_length=30)

```

views.py

```

def home(request):
    return render(request, 'home.html')

def registration(request):
    if request.method == "POST":
        form = RegistrationForm(request.POST)
        if form.is_valid():
            form.save()
            print("valid Details")
            return redirect('login')
        else:
            form = RegistrationForm()
            print("Invalid Details")
            return render(request, 'registration.html', {'form':form})

def payment(request):
    if request.method=="POST":
        form=PaymentForm(request.POST)
        if form.is_valid():
            form.save()
            return redirect('pay')
        else:
            form=PaymentForm(request.POST)
            return render(request,'payment.html',{'form':form})
    return render(request,'payment.html',{'form':form})

```

models.py

```

class Registration(models.Model):
    username = models.CharField(max_length=30)
    email = models.CharField(max_length=30)

```

```
password = models.CharField(max_length=30)
dob=models.CharField(max_length=30)
contact=models.CharField(max_length=30)
```

```
class itemsdonation(models.Model):
    itemtype=models.CharField(max_length=30)
    shopname=models.CharField(max_length=30)
    shopmobno=models.CharField(max_length=30)
    deliveryadd=models.CharField(max_length=30)
```

```
class Payment(models.Model):
    dname=models.CharField(max_length=30)
    demail=models.CharField(max_length=30)
    dmobno=models.CharField(max_length=30)
    damt=models.CharField(max_length=30)
    orphansname=models.CharField(max_length=30)
```

urls.py

```
urlpatterns = [
    url(r'^admin/', admin.site.urls),
    url(r'^$', views.home,name='home'),
    url(r'^home/', views.home,name='home'),
    url('registration/', views.registration,name='registration'),
    url('login/', views.login,name='login'),
    url('itemsdonation/', views.itemsdonation,name='itemsdonation'),
    url('payment/', views.payment,name='payment'),
    url('pay/', views.pay,name='pay'),
]
```

5. Test Results

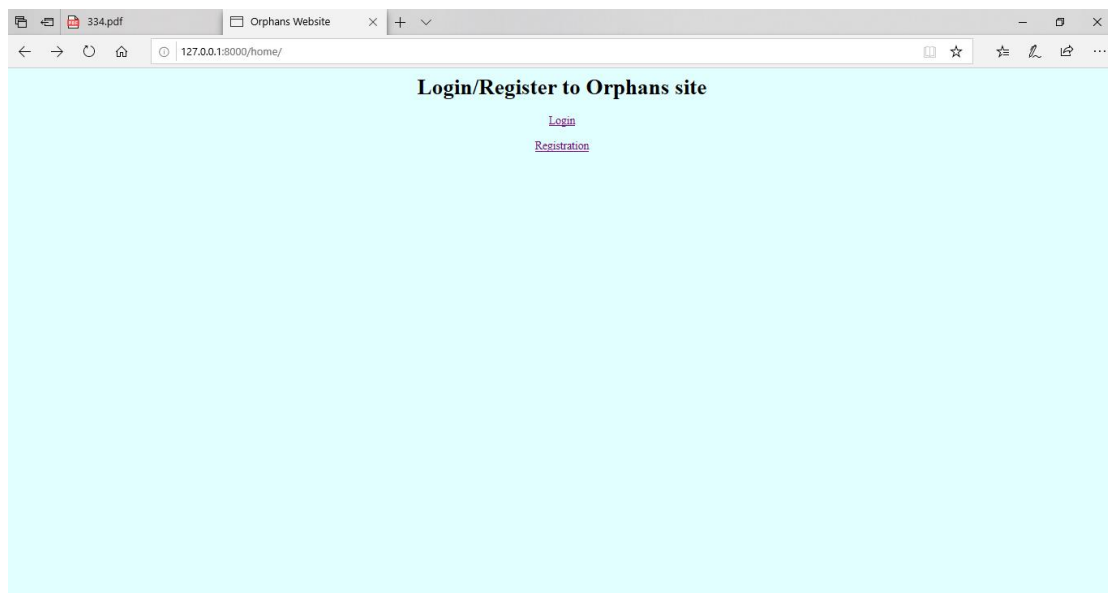
5.1 Test Cases

S.NO	PROBLEM STATEMENT	INPUT	OUTPUT	EXPECTED OUTPUT
1	Login	Enter username and password	Go to item donation page.	Redirect to Item donation page.
2	Login	Enter valid username and invalid password	Invalid credentials	Invalid credentials
3	Login	Enter invalid username and invalid password	Invalid credentials	Invalid credentials
4	Login	Enter invalid username and valid password	Invalid credentials	Invalid credentials
5	Registration	Enter all valid details like user id, email and contact	Create a User ID	Create User ID
6	Registration	Enter any one invalid detail.	Enter valid credentials	Enter Valid credentials
7	Item Donation	Enter details of donor and vendor.	Update the details in database.	Update the details in database.
8	Item Donation	Submit the details	Redirect to payment page.	Redirect to payment page.
9	Item Donation	Submit without entering details	This is a required field	This is a required field
10	Payment	Enter Donor details, amount, orphan details and payment method	Redirect to pay page.	Redirect to pay page.

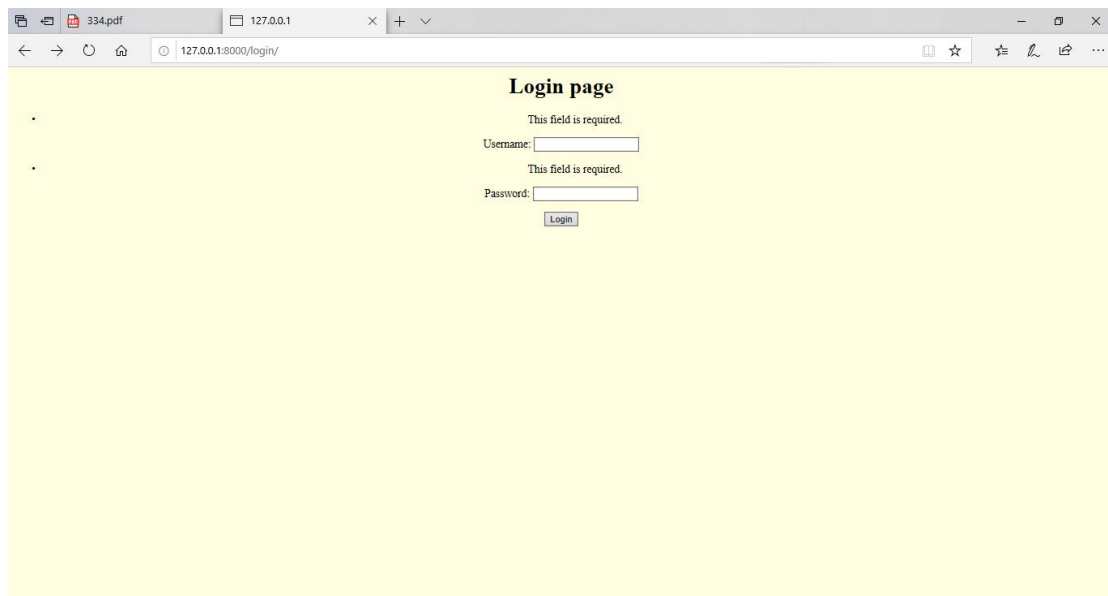
11	Payment	Keeping a section blank	This is a required field	This is a required field
12	Pay	Enter card details	Complete the transaction.	Complete the transaction.
13	Pay	Do not enter card details	This is a required field.	This is a required field.
14	Pay	Enter card number as alphabets.	Enter valid details.	Enter valid details.
15	Payment	Enter invalid details	Check the details	Check the details

6. Results and Discussions

Home Page



Login Page



334.pdf 127.0.0.1

127.0.0.1:8000/login/

Login page

This field is required.

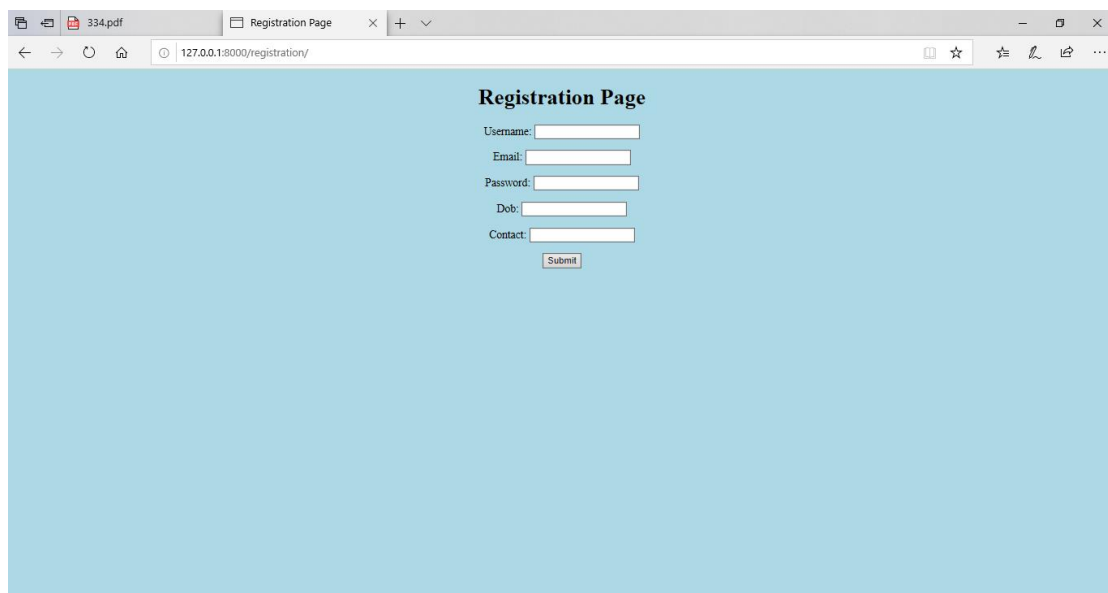
Username:

This field is required.

Password:

Login

Registration Page:



334.pdf Registration Page

127.0.0.1:8000/registration/

Registration Page

Username:

Email:

Password:

Dob:

Contact:

Submit

Item Donation Page:

The screenshot shows a web browser window with the address bar displaying "127.0.0.1:8000/itemsdonation/". The page has a light blue background and is titled "Items Donation Page" in bold black text. On the left side, there is a vertical list of seven bullet points. To the right of these points is a form with the following fields and labels:

- Itemtype: [text input] (This field is required.)
- Shopname: [text input] (This field is required.)
- Shopmobno: [text input] (This field is required.)
- Deliveryadd: [text input] (This field is required.)
- Select the States: [dropdown menu] (This field is required.)
- Select the Areas: [dropdown menu] (This field is required.)

At the bottom of the form is a red "submit" button.

Payment Details:

The screenshot shows a web browser window with the address bar displaying "127.0.0.1:8000/payment/". The page has a light blue background and is titled "Payment Details" in bold black text. On the left side, there is a vertical list of seven bullet points. To the right of these points is a form with the following fields and labels:

- Dname: [text input] (This field is required.)
- Demail: [text input] (This field is required.)
- Dmobno: [text input] (This field is required.)
- Damt: [text input] (This field is required.)
- Orphansname: [text input] (This field is required.)
- Select the Payments: [dropdown menu] (This field is required.)

At the bottom of the form is a red "payment" button.

Pay Details:

Pay Details

• This field is required.
Cardholder:

• This field is required.
Cardnum:

• This field is required.
Cvv:

• This field is required.
Expdate:

Database Screenshots:

```
MySQL 5.5 Command Line Client
Enter password: ****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 6
Server version: 5.5.62 MySQL Community Server (GPL)

Copyright (c) 2000, 2018, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| orphdb |
| performance_schema |
| test |
+-----+
5 rows in set (0.11 sec)

mysql> use orphdb
Database changed
mysql> show tables;
+-----+
| Tables_in_orphdb |
+-----+
| django_migrations |
| orphansapp_itemsdonation |
| orphansapp_payment |
| orphansapp_registration |
+-----+
4 rows in set (0.00 sec)
```

```

mysql> select * from orphansapp_itemsdonation;
+-----+-----+-----+-----+-----+
| id | itemtype | shopname | shopmobno | deliveryadd |
+-----+-----+-----+-----+-----+
| NULL | books | bookmybooks | 123456789 | Hyderabad |
| NULL | Pens | Sai Stationary | 987456321 | Koti |
| NULL | chocolates | Krishna Bakery | 9985632147 | Ramnagar |
| NULL | clothes | RS Brothers | 9952885524 | Dilshuknagar |
+-----+-----+-----+-----+-----+
4 rows in set (0.03 sec)

mysql> select * from orphansapp_registration;
+-----+-----+-----+-----+-----+-----+
| id | username | email | password | dob | contact |
+-----+-----+-----+-----+-----+-----+
| NULL | kiran | skn@gmail.com | 1234 | 19061998 | 987456321 |
| NULL | sai | sai@gmail.com | 4567 | 19061998 | 9632587410 |
+-----+-----+-----+-----+-----+-----+
2 rows in set (0.00 sec)

mysql>

```

7. Conclusion and Future Work

7.1 Conclusion

This system provides a platform to the welfare of the orphanage and can be used by NGOs for co-operative social work. The user can register here and carry on their donation work by viewing the details of the orphanage. It improves the efficiency to the user towards their social work.

7.2 Future Work

Enhancements that can be included in this system are Orphanages all across India can be included, Notifications via SMS can be done for donation and registration Education and medical details can be added to children details.

We can create a mobile based application similar to the web application so that user can access the service from any location.

8. REFERENCES

[1]Haruna, Muawiyya Adam, Yashi, Kamaluddeen Musa, Zakiya, YahayaShehu, and Abdulganiyu, Sumaiya, "Design and Development of an Orphans Record System", Volume 8, Number 2, December, 2017

[2]M.Archana¹, K.Mouthami², “Charity Connecting System”, Volume III, Issue VII, July 2014

[3] George, N. (2018). Build your first website with Django 2.1: Master the basics of Django while building a fully-functioning website.