

CROWD-FUNDING USING BLOCKCHAIN

A Mini Project Report

submitted by

Rinsha Ap(MES20MCA-2039)

to

the APJ Abdul Kalam Technological University
in partial fulfillment of the requirements for the award of the Degree

of

Master of Computer Applications



Department of Computer Applications

MES College of Engineering
Kuttippuram, Malappuram - 679 582

February 2022

DECLARATION

I undersigned hereby declare that the project report **crowd-funding using blockchain**, submitted for partial fulfillment of the requirements for the award of degree of Master of Computer Applications of the APJ Abdul Kalam Technological University, Kerala, is a bonafide work done by me under supervision of Ms.Priya.J.D, Assistant Professor, Department of Computer Applications. This submission represents my ideas in my own words and where ideas or words of others have been included, I have adequately and accurately cited and referenced the original sources. I also declare that I have adhered to ethics of academic honesty and integrity and have not misrepresented or fabricated any data or idea or fact or source in my submission. I understand that any violation of the above will be a cause for disciplinary action by the institute and/or the University and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been obtained. This report has not been previously formed the basis for the award of any degree, diploma or similar title of any other University.

Place: Kuttippuram

Date: 28/02/2022

Rinsha Ap (MES20MCA-2039)

DEPARTMENT OF COMPUTER APPLICATIONS
MES COLLEGE OF ENGINEERING, KUTTIPPURAM



CERTIFICATE

This is to certify that the report entitled **crowd-funding using blockchain** is a bonafide record of the Mini Project work carried out by **Rinsha Ap(MES20MCA-2039)** submitted to the APJ Abdul Kalam Technological University, in partial fulfillment of the requirements for the award of the Master of Computer Applications, under my guidance and supervision. This report in any form has not been submitted to any other University or Institution for any purpose.

Internal Supervisor(s)

External Supervisor(s)

Head Of The Department

Acknowledgements

At the very outset I would like to thank the almighty's mercy towards me over the years. I wish to express my sincere thanks to my project guide Ms.Priya J.D Assistant Professor, Dept. of Master of Computer Applications who guided me for the successful completion of this project. I also thank her for valuable suggestions, guidance, constant encouragement, boundless corporation, constructive comments and motivation extended to me for completion of this project work. I would express my sincere thanks to Mr.Hyderali K, Associate Professor and Head of Department for his immense guidance to complete the project successfully. I would like to express my sincere thanks to all the faculty members of Master of Computer Applications department for their support and valuable suggestion for doing the project work. Last but not least my graceful thanks to my parents, friends and also the persons who supported me directly and indirectly during the project.

Rinsha Ap(MES20MCA-2039)

Abstract

There are lots of developments in the fields of science and technology but even in these days very important agreements related to properties, Crowdfunding are made on paper which is not at all safe as they can be duplicated or forgery can be done. Even if the agreements are digital they can be hacked if no security surrounds them. But if agreements, applications are made using smart contracts the security will be increased by a large scale. In normal crowd funding the entire amount is directly in the hands of manager and also the amount used by manager doesn't involve investors opinion. So, there is no security ensured in this case. The proposed methodology creates a smart contract in such a way that the entire funding amount will be inside the smart contract and when the manager wants to make use of funding amount and buy something from a vendor, he/she has to make a spending request and then investors have to vote on that request. If the contract gets majority amount of votes the amount will be automatically sent to the vendor. In this way process of crowd funding can be made better and trustworthy.

Keywords:Blockchain, Smart Contracts, Crowdfunding, etc

Contents

Declaration	i
Certificate	ii
Acknowledgements	iii
Abstract	iv
Contents	v
List of Figures	vi
List of Tables	viii
1 Introduction	1
1.1 Background	1
1.2 Objective	2
1.2.1 Motivation	2
1.3 Report Organization	3
2 Literature Survey	4
3 Methodology	5
3.1 Introduction	5
4 Conclusions	7
References	8
Appendix	9
Source Code	9
4.1 Results	20
4.2 Project Plan	29
4.3 Product Backlog	30
4.4 Sprint Plan	31
4.5 Sprint Actual	32

List of Figures

A.1	Dataflow Diagram	18
A.2	Dataflow Diagram	18
A.3	Dataflow Diagram	19
A.4	login page	20
A.5	Admin login page	20
A.6	Admin home page	21
A.7	Manger login page	21
A.8	Manager home page	22
A.9	Create projects	22
A.10	Sent request	23
A.11	View Stakeholder	23
A.12	login stakeholder	24
A.13	stakeholder home page	24
A.14	View projects	25
A.15	Vote for request	25
A.16	send request	26
A.17	view request status	26
A.18	view project funt	27
A.19	send request	27
A.20	view accepted projects	28
A.21	Project Plan	29
A.22	Project Plan	29

LIST OF FIGURES

vii

A.23 Product Backlog 30

A.24 Product backlog 30

A.25 Sprint Plan 31

A.26 Sprint Plan 31

A.27 Sprint Actual 32

A.28 Sprint Actual 32

List of Tables

A.1	Bank	15
A.2	cipher	15
A.3	Login	15
A.4	Project	16
A.5	Project _a llocation	16
A.6	Pvalet	16
A.7	Request	16
A.8	17
A.9	valet	17
A.10	vote	17

Chapter 1

Introduction

1.1 Background

Crowdfunding is the practice of funding a project or venture by raising small amounts of money from a large number of people, typically via the internet. Before crowdfunding were started people would really struggle to get funding for startups. It was difficult for people to convince investors to invest in their ideas. But now there is no need of such struggle as many crowdfunding platforms have evolved like Kickstarter, Indiegogo etc. which are helping people for getting the required funding and thereby making their dreams come true. In the existing system of crowdfunding platform like Kickstarter in order to list any campaign, the campaign creator should give details like the idea of their project, funding goal, government issued ID proof, credit/debit card details etc. The project was developed using Agile Development Model. The entire project was divided into four sprints. In the first sprint, basic coding, form design and table design was done. In the second sprint, view project current status, registration of stakeholders was done. In third sprint view project information from manager, send fund amount, view project fund and spending request was done. And in the last sprint accept/reject spending request and test data was done also document upload, verification, sanctioning of claim and view claim status was done.

1.2 Objective

The details of the admin and manager are stored in the database. The project is based on web and android application. the web is maintained by the admin and manager. the next part is android it is used by the stakeholder. The stakeholder can view projects, vote for request, request status. The main objective of our project is to overcome the problem of security and we are dealing it by using the technology of smart contract which is an application of blockchain. In the proposed system the funded amount is kept inside a smart contract and the amount does not directly goes to creator whereas it goes to a vendors address specified by the campaign creator. Before that a spending request must be generated to spend the money kept inside the contract. Within the request the campaign creator need to specify the vendors address, purpose of the spending request and how much money is to be released from the contract. After creating the spending request, investors are given a chance to show their opinion by casting vote whether to sanction the spending request. Approval of majority is mandatory, that 51

1.2.1 Motivation

The voting system here is decentralized as blockchain technology is used in implementing it. This makes it more secure and also cost efficient while guaranteeing the voters privacy. For solving the limitations of the existing system an administrator module is assigned to verify the identity of the users login in to the platform and extra security is given so that to verify user profiles are real and project ideas need to be verified by the admin first to ensure its not scam. In this the security is increased and also the peoples/contributors opinion is taken. It also ensures there is proper communication between the investors and the creators. Here the smart contract is written in a way that ensures the backers would get the benefit of the investing in this project by keeping a fixed amount of money in the contract as backer insurance so that it would not be lost at any context.

1.3 Report Organization

The project report is divided into four sections. section two describes literature survey. section three describes the methodology used for implementing the project. section four gives the conclusion.

Chapter 2

Literature Survey

Crowdfunding has a long history with several roots. Books have been crowdfunded for centuries; authors and publishers would advertise book projects in praenumeration or subscription schemes. The book would be written and published if enough subscribers signaled their readiness to buy the book once it was out. The subscription business model is not exactly crowdfunding, since the actual flow of money only begins with the arrival of the product. The list of subscribers has, though, the power to create the necessary confidence among investors that is needed to risk the publication.

War bonds are theoretically a form of crowdfunding military conflicts. London's mercantile community saved the Bank of England in the 1730s when customers demanded their pounds to be converted into gold - they supported the currency until confidence in the pound was restored, thus crowdfunded their own money. A clearer case of modern crowdfunding is Auguste Comte's scheme to issue notes for the public support of his further work as a philosopher. The "Première Circulaire Annuelle adressée par l'auteur du Système de Philosophie Positive" was published on March 14, 1850, and several of these notes, blank and with sums have survived

Chapter 3

Methodology

3.1 Introduction

Crowd funding is the practice of funding a project or venture by raising small amounts of money from a large number of people, typically via the internet. Before crowd funding were started people would really struggle to get funding for startups. It was difficult for people to convince investors to invest in their ideas. But now there is no need of such struggle as many crowdfunding platforms have evolved like Kick starter, Indiegogo etc. which are helping people for getting the required funding and thereby making their dreams come true. The investors who are interested will invest in these ideas and if the amount reaches the desired goal within the deadline, the entire amount excluding some middlemen fee will be given to campaign creator. The main problem faced in this system is that there is no ensured security, the whole funding amount is given to the person who created the campaign. There are chances that this may lead to miscellaneous activities. People with intention to make money easily will use this platform in a wrong way by coming up with fake ideas and the investors will end up losing all the money invested. Even though there are digital contracts available, they are not ensuring the investor's interest or security. So, investing in these kinds of platforms is highly risky and depends on luck. So, our project focuses on increasing the security with the help of blockchain technology and smart contracts, which makes the transactions transparent for the investors and it ensures the funding amount will be safe inside these contracts.

Here if the campaign creator wants to spend the fund, he has to create a spending request specifying the reason to spend the fund, to whom the amount goes to (vendors address) and, how much fund is to be released. The investors can vote on this request by approving or can reject the request. Only if the majority is supporting the request the fund is able to release. Here the amount is not directly giving to the campaign creator, but to the vendors address specified in the spending request. The proposed system is a solution to overcome the issues with the existing system. This project was developed using Agile Development Model. The entire project was divided into four sprints. In the first sprint, the characters for the password was developed. The designing of front-end and development of back-end was done in the second , third and fourth sprint respectively.

The goal of blockchain is to allow digital information to be recorded and distributed, but not edited. In this way, a blockchain is the foundation for immutable ledgers, or records of transactions that cannot be altered, deleted, or destroyed. Blockchain technologies are the digital, distributed, and decentralized ledger representing the most virtual currencies that are accountable for logging all transactions without the need for a financial mediator, such as a bank. In other words, it's a new means of transmitting funds and logging information

RSA ALGORITHM: The RSA algorithm is an asymmetric cryptography algorithm; this means that it uses a public key and a private key (i.e two different, mathematically linked keys). As their names suggest, a public key is shared publicly, while a private key is secret and must not be shared with anyone. The RSA algorithm involves four steps: key generation, key distribution, encryption, and decryption

SHA256: The SHA-256 algorithm is one flavor of SHA-2 (Secure Hash Algorithm 2), which was created by the National Security Agency in 2001 as a successor to SHA-1. SHA-256 is a patented cryptographic hash function that outputs a value that is 256 bits long.

Chapter 4

Conclusions

The paper proposed a smart contract based solution to enable secure way of crowd funding by ensuring that the money donated by the investors is safe and also each and every step taken in the startup with help of donated money involves investor's opinion i.e. whenever the campaign creator wants to spend the money he/she has to make a spending request where the purpose of using the money, to whom the money is being sent(vendor) and the amount needed should be mention. The main advantage of using the smart contract which is a concept of blockchain is that it is resilient against many threats. Also it provides many features like improved reliability, faster and efficient operation. Web Application has been computed successfully and was also tested by taking "test cases". It is user friendly, and has required options, which can be utilized by user to perform the desired operation. The goals achieved by the software are: Creating a campaign Contributing to campaign Creating a spending request Approving the spending request Finalising payment Limitations regarding the project are that the vendor address is not verified by which campaign creator and the vendor can get together and scam the contributors. Also one person can contribute only one time for campaign/startup from a account as of now. The Ethereum accounts from which the contributors are investing also not verified. Future works will aim to find a way to verify the Ethereum accounts whether it may be Metamask or MyEtherWallet(Ethereum wallet similar to metamask) accounts with help of a government id so that one person cannot have more than one account and also make sure that a person is able to contribute as many times he/she wants, so that the Crowdfunding using blockchain becomes more secure and reliable.

References

- [1] E. Mollick, The dynamics of crowdfunding: An exploratory study, Journal of business venturing, vol. 29, no. 1, pp. 1-16, Jan, 2014.
- [2] M. E. Peck, "Blockchains: How they work and why they'll change the world," in IEEE Spectrum, vol. 54, no. 10, pp. 26-35, 2017
- [3] VitalikButerin, A next-generation smart contract and decentralized application platform. white paper ,2014
- [4] PéterHegedus, Towards Analysing the Complexity Landscape of Solidity Based Ethereum Smart Contracts, Proc. 1st IEEE International Workshop on Emerging Trends in Software Engineering for Blockchain, Gothenburg, Sweden, 2018.

Appendix

Source Code

```
# This piece of software is bound by The MIT License (MIT)
# Copyright (c) 2013 Siddharth Agrawal
# Code written by : Siddharth Agrawal
# Email ID : siddharth.950@gmail.com

import os
from flask import *
from werkzeug.utils import secure_filename
from src.dbconnector import *
app=Flask(__name__)
app.secret_key="secure"
import functools
def login_required(func):
    @functools.wraps(func)
    def secure_function():
        if "ln" not in session:
            return redirect("/")
        return func()
    return secure_function
# @app.route('/')
# def main():
#     return render_template('index.html')
@app.route('/')
def log():
    return render_template('login.html')
@app.route('/logout')
def logout():
    session.clear()
    return render_template('login.html')
@app.route('/login',methods=['post'])
def login():
    uname=request.form['textfield']
    pword=request.form['textfield2']
    qry="SELECT * FROM 'login' WHERE 'username'=%s AND 'password'=%s"
    val=(uname,pword)
    res=selectone(qry,val)
    if res is None:
        return '''<script>alert('invalid enrtty');window.location="/"</script>'''
    else:
        if res[3]=='admin':
            session['ln']=res[0]
            return '''<script>alert('welcome');window.location="/adminhome"</script>'''
        elif res[3] == 'manager':
            session['ln']=res[0]
```

Appendix

```
        return '''<script>alert('welcome');window.location="/managerhome"</script>'''
    else:
        return '''<script>alert('invalid entry');window.location="/"</script>'''
@app.route('/addmanagers')
def addmanagers():
    return render_template('addmanagers.html')
@app.route('/acceptmanagers')
@login_required
def acceptmanagers():
    id=request.args.get('id')
    q="update 'login' set type='manager' WHERE 'id'=%s"
    v=str(id)
    iud(q,v)
    # qry="DELETE FROM 'manager' WHERE 'loginid'=%s"
    # iud(qry,v)
    return '''<script>alert('Accepted');window.location="/managemanagers"</script>'''
@app.route('/rjctmanagers')
@login_required
def rjctmanagers():
    id=request.args.get('id')
    q="update 'login' set type='reject' WHERE 'id'=%s"
    v=str(id)
    iud(q,v)
    return '''<script>alert('Rejected');window.location="/managemanagers"</script>'''
@app.route('/dltsh')
@login_required
def dltsh():
    id=request.args.get('id')
    q="DELETE FROM 'login' WHERE 'id'=%s"
    v=str(id)
    iud(q,v)
    qry="DELETE FROM 'stockholder' WHERE 'loginid'=%s"
    iud(qry,v)
    return '''<script>alert('deleted');window.location="/managestockholders"</script>'''
@app.route('/dltproj')
@login_required
def dltproj():
    id=request.args.get('id')
    v=str(id)
    qry="DELETE FROM 'project' WHERE 'pro_id'=%s"
    iud(qry,v)
    return '''<script>alert('deleted');window.location="/createproject"</script>'''
@app.route('/dltreq')
@login_required
def dltreq():
    id=request.args.get('id')
    v=str(id)
    qry="DELETE FROM 'request' WHERE 'id'=%s"
    iud(qry,v)
    return '''<script>alert('deleted');window.location="/sendrequest1"</script>'''
@app.route('/accept')
@login_required
def accept():
    id=request.args.get('id')

    return render_template("adminlastdate.html",val=id)

@app.route('/accept1',methods=['post'])
@login_required
def accept1():
    date=request.form['date']
    pid=request.form['id']
    qry="insert into lastdate values(null,%s,%s)"
```

Appendix

```
v=(pid,date)
iud(qry,v)
v = str(pid)
qry = "update `project` set status='requested' WHERE `pro_id`=%s"
iud(qry, v)
return '''<script>alert('accepted');window.location="/adminmanageprojects"</script>'''

@app.route('/reject')
@login_required
def reject():
    id=request.args.get('id')
    v=str(id)
    qry="update `project` set status='reject' WHERE `pro_id`=%s"
    iud(qry,v)
    return '''<script>alert('rejected');window.location="/adminmanageprojects"</script>'''
@app.route('/addingmanager',methods=['post'])
def addingmanager():
    try:
        fname=request.form['textfield7']
        lname=request.form['textfield6']
        dob=request.form['textfield5']
        gend=request.form['radiobutton']
        pno=request.form['textfield4']
        email=request.form['textfield3']
        qualif=request.form.getlist('checkbox')
        qua=', '.join(qualif)
        uname=request.form['textfield2']
        pwd=request.form['textfield']
        pic=request.files['file']
        qry="INSERT INTO `login` VALUES(NULL,%s,%s,'pending')"
        val=(uname,pwd)
        res=iud(qry,val)
        path1 = "./static/pic/" + str(res) + ".png"
        pic.save(path1)
        q="INSERT INTO `manager` VALUES(NULL,%s,%s,%s,%s,%s,%s,%s)"
        v=(str(res),fname,lname,dob,gend,pno,email,qua)
        iud(q,v)
        return '''<script>alert('registration successfull');window.location="/"</script>'''
    except:
        return jsonify({'task': 'username already exists'})
@app.route('/stockholders',methods=['post'])
@login_required
def stockholders():
    try:
        fname = request.form['textfield']
        lname = request.form['textfield2']
        plc = request.form['textfield3']
        post = request.form['textfield4']
        pin = request.form['textfield5']
        pno = request.form['textfield6']
        email = request.form['textfield7']
        uname = request.form['textfield8']
        pwd = request.form['textfield9']
        qry = "INSERT INTO `login` VALUES(NULL,%s,%s,'stakeholder')"
        val = (uname, pwd)
        res = iud(qry, val)
        q = "INSERT INTO `stockholder` VALUES(NULL,%s,%s,%s,%s,%s,%s,%s)"
        v = (str(res), fname, lname, plc, post, pin, email, pno)
        iud(q, v)
        return '''<script>alert('registration successfull');window.location="/managestockholders"</script>'''
    except:
```

Appendix

```
        return jsonify({'task': 'username already exists'})
@app.route('/addstockholders',methods=['post'])
@login_required
def addstockholders():
    return render_template('addstockholders.html')
@app.route('/adminhome')
@login_required
def adminhome():
    return render_template('adminhome.html')
@app.route('/addproj',methods=['post'])
@login_required
def addproj():
    proj = request.form['textfield']
    des = request.form['textfield2']
    amt = request.form['textfield3']
    det = request.files['file']
    fn=secure_filename(det.filename)
    det.save(os.path.join('static/files',fn))
    q = "INSERT INTO 'project' VALUES(NULL,%s,%s,%s,%s,%s,'pending',curdate())"
    v = ( str(session['ln']),proj, des, amt,fn)
    iud(q,v)
    return '''<script>alert('success');window.location="/sendrequest1"</script>'''
@app.route('/adminmanageprojects')
@login_required
def adminmanageprojects():
    qry = "SELECT 'project'.*, 'manager'. 'fname', 'manager'. 'lname' FROM 'project' JOIN 'manager' ON
        'project'.mid='manager'. 'loginid' WHERE 'project'. 'status'='pending'"
    s = select(qry)
    return render_template('adminmanageprojects.html',v=s)
@app.route('/createproject')
@login_required
def createproject():
    qry = "SELECT project.*, 'lastdate'. 'lastdate' FROM 'project' left JOIN 'lastdate' ON 'lastdate'. 'pid'='project'. 'pro_id'
        where 'project'.mid='"+str(session['ln'])+"'"
    s = select(qry)
    return render_template('createproject.html',v=s)
@app.route('/cproj',methods=['post'])
@login_required
def cproj():
    proj = request.form['slt']
    rsn = request.form['textfield3']
    amt = request.form['textfield2']
    q = "INSERT INTO 'request' VALUES(NULL,%s,%s,%s,curdate(),'pending')"
    v = ( proj,amt,rsn)
    iud(q,v)
    return '''<script>alert('requested');window.location="/sendrequest1"</script>'''
@app.route('/createproject2',methods=['post'])
@login_required
def createproject2():
    return render_template('createproject2.html')
@app.route('/managemanagers')
@login_required
def managemanagers():
    qry="SELECT 'manager'.* FROM 'manager' JOIN 'login' ON 'manager'. 'loginid'='login'. 'id' WHERE 'login'.type='pending'"
    s=select(qry)
    return render_template('managemanagers.html',v=s)
@app.route('/managerhome')
@login_required
def managerhome():
    return render_template('managerhome.html')
@app.route('/managestockholders')
@login_required
def managestockholders():
```

Appendix

```
qry = "SELECT 'stockholder'.* FROM 'stockholder' JOIN 'login' ON 'stockholder'.loginid='login'.id WHERE
      'login'.type='pending'"
s = select(qry)
return render_template('managestockholders.html',v=s)
@app.route('/acceptsh')
@login_required
def acceptsh():
    id=request.args.get('id')
    q="update 'login' set type='stakeholder' WHERE 'id'=%s"
    v=str(id)
    iud(q,v)
    return '''<script>alert('Accepted');window.location="/managestockholders"</script>'''
@app.route('/sendrequest1')
@login_required
def sendrequest1():
    q="SELECT 'request'.*, 'project'. 'project', 'project'. 'description' FROM 'project' JOIN 'request' ON
      'project'. 'pro_id'='request'.pid where 'project'. 'mid'='"+str(session['ln'])+"'"
    s=select(q)
    return render_template('sendrequest1.html',v=s)
@app.route('/sendrequest2',methods=['post'])
@login_required
def sendrequest2():
    q="select * from project where status='accept'"
    s=select(q)
    return render_template('sendrequest2.html',val=s)
@app.route('/viewstatus')
@login_required
def viewstatus():
    con = pymysql.connect(host='localhost', port=3306, user='root', passwd='', db='cloudfunding')
    cmd = con.cursor()
    s=[]
    cmd.execute("SELECT 'project'. 'pro_id', 'request'. 'id', 'project'. 'project', 'reason', 'request'. 'amount' FROM 'request'
      JOIN 'project' ON 'project'. 'pro_id'='request'. 'pid' WHERE 'project'. 'mid'='"+str(session['ln'])+"")
    ss=cmd.fetchall()
    for i in ss:
        cmd.execute("SELECT COUNT(*) FROM 'project_allocation' WHERE 'pid'='"+str(i[0])")
        stcount=cmd.fetchone()
        cmd.execute("SELECT * FROM 'vote' WHERE 'rid'='"+str(i[1])")
        swcount=cmd.fetchall()
        row=[]
        row.append(i[2])
        row.append(i[3])
        row.append(i[4])
        if(stcount[0] != 0):
            p=len(swcount)/int(stcount[0])
            p=p*100
            row.append(p)
        else:
            p= 0
            row.append(p)
        row.append(int(stcount[0])-len(swcount))
        s.append(row)
    return render_template('viewstatus.html', val=s)
@app.route('/viewstockholders')
@login_required
def viewstockholders():
    qry = "SELECT 'stockholder'.*, 'project'. 'project' FROM 'stockholder' JOIN 'project_allocation' ON
          'project_allocation'. 'shid' ='stockholder'. 'loginid' JOIN 'project' ON
          'project_allocation'. 'pid'='project'. 'pro_id' where 'project'. 'mid'='"+str(session['ln'])+"'"
    s = select(qry)
    return render_template('viewstockholders.html', v=s)
@app.route('/myvalet')
@login_required
```

Appendix

```
def myvalet():
    id=str(session['ln'])
    v=str(id)
    qry="SELECT 'manager'.fname, 'manager'.lname, 'manager'.email, 'valet'.balance FROM 'manager' LEFT JOIN 'valet' ON
        'manager'.loginid='valet'.uid WHERE 'manager'.loginid=%s"
    res=selectone(qry,v)
    return render_template("myvv.html",v=res)
@app.route('/pvalet')
@login_required
def pvalet():
    id=request.args.get('id')
    v=str(id)
    qry="SELECT 'project'.project, 'project'.amount, 'project'.date, 'pvalet'.balance FROM 'project' LEFT JOIN 'pvalet'
        ON 'project'.pro_id='pvalet'.id WHERE 'project'.pro_id=%s"
    res=selectone(qry,v)
    return render_template("projectv.html",v=res)
app.run(debug=True)
```

Database Design

Insert the database design here (if any). A sample format is given below

Attribute Name	Datatype	Width	Description
id	Integer	11	NOT NULL
uid	int	11	Null
account _{no}	varchar	600	NULL
ifsc	varchar	600	NULL
pin	bigint	20	Null
account	float		NULL
bank	varchar	500	NULL

Table A.1: Bank

Attribute Name	Datatype	Width	Description
id	Int	11	NOT NULL
public _{key}	text		NULL
privacy _{key}	text		NULL

Table A.2: cipher

Attribute Name	Datatype	Width	Description
id	Int	11	NOT NULL
username	varchar	40	NULL
password	varchar	40	NULL
type	varchar	40	NULL

Table A.3: Login

Attribute Name	Datatype	Width	Description
pro_id	Integer	11	NOT NULL
mid	int	11	Null
project	varchar	50	NULL
amount	bigint	20	NULL
details	varchar	100	Null
status	varchar	20	NULL
date	varchar	20	NULL

Table A.4: Project

Attribute Name	Datatype	Width	Description
id	Int	11	NOT NULL
pid	int	11	Null
shid	int	11	NULL
amount	bigint	20	NULL
date	varchar	20	NULL

Table A.5: Project_{allocation}

Attribute Name	Datatype	Width	Description
id	Int	11	NOT NULL
balance	float		Null

Table A.6: Pvalet

Attribute Name	Datatype	Width	Description
id	Int	11	NOT NULL
pid	int	11	Null
amount	varchar	30	NULL
reason	varchar	700	Null
status	varchar	20	NULL
date	varchar	20	NULL

Table A.7: Request

Attribute Name	Datatype	Width	Description
id	Int	11	NOT NULL
loginid	int	11	Null
fname	varchar	50	NULL
iname	varchar	50	NULL
place	varchar	50	Null
post	varchar	50	NULL
pin	bigint	20	NULL
email	varchar	50	NULL
contact	bigint	20	NULL
banck	varchar	100	Null
account _{no}	varchar	50	NULL
ifac	bigint	20	NULL

Table A.8:

Attribute Name	Datatype	Width	Description
id	Int	11	NOT NULL
uid	Int	11	NOT NULL
balance	float		Null

Table A.9: valet

Attribute Name	Datatype	Width	Description
id	Int	11	NOT NULL
rid	Int	11	NOT NULL
shid	int	11	Null
vote	varchar	20	Null

Table A.10: vote

DaTaflow Diagram

Keep your dataflow diagram here (if any).

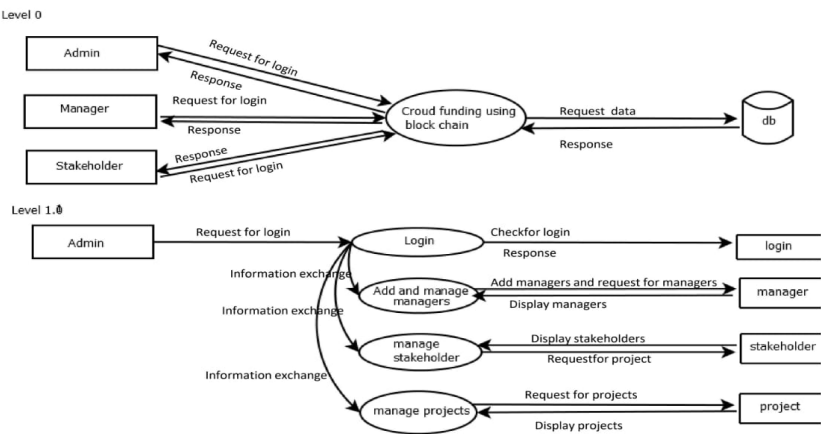


Figure A.1: Dataflow Diagram

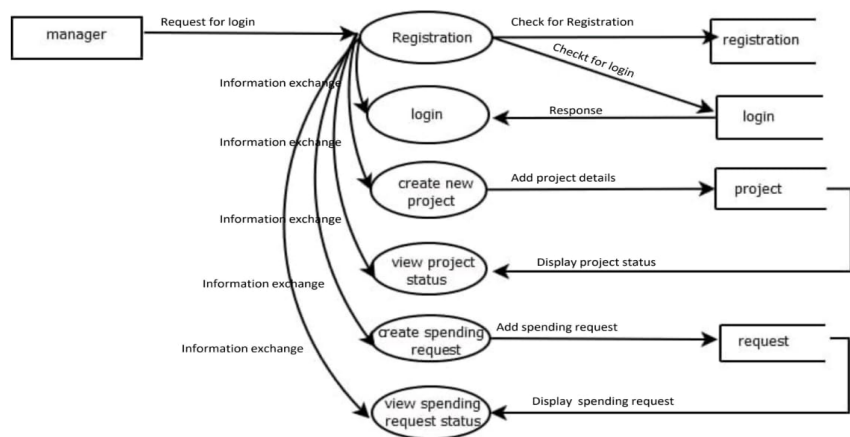


Figure A.2: Dataflow Diagram

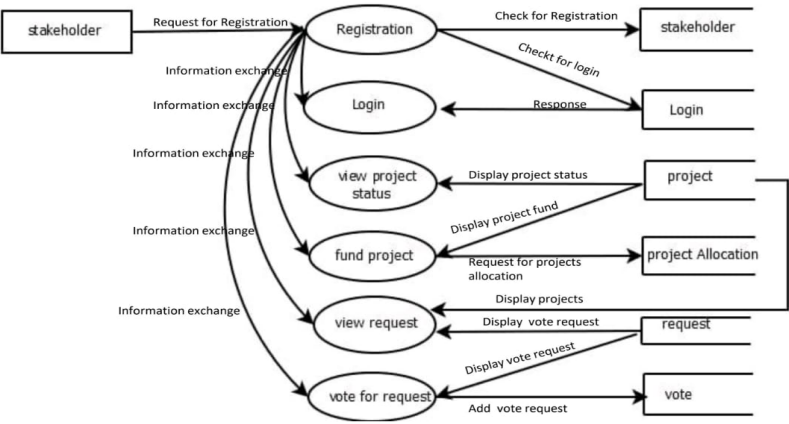


Figure A.3: Dataflow Diagram

4.1 Results

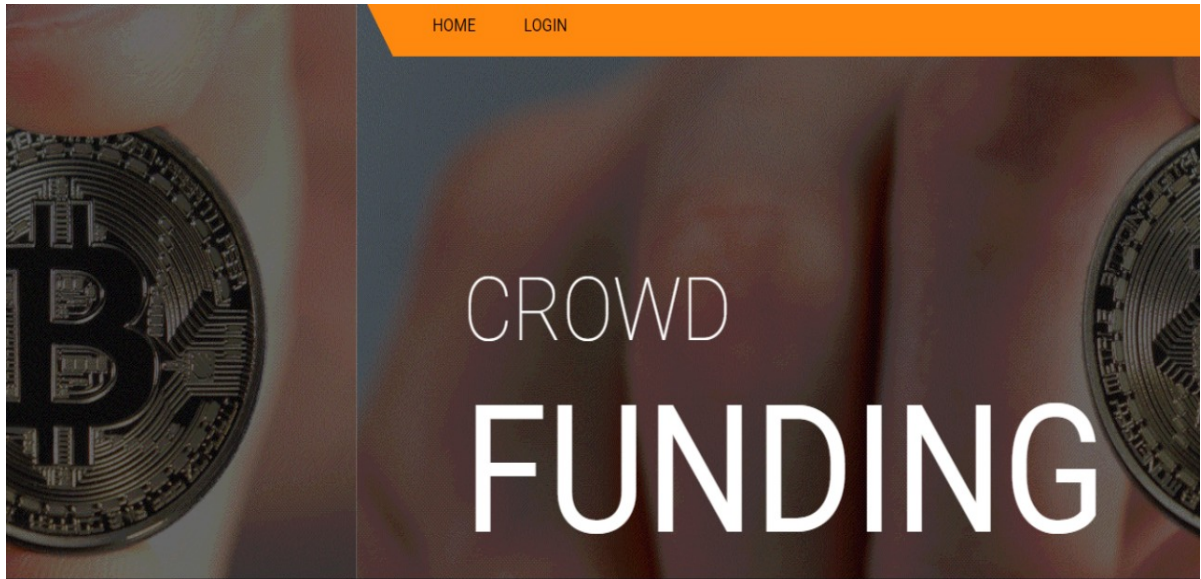


Figure A.4: login page

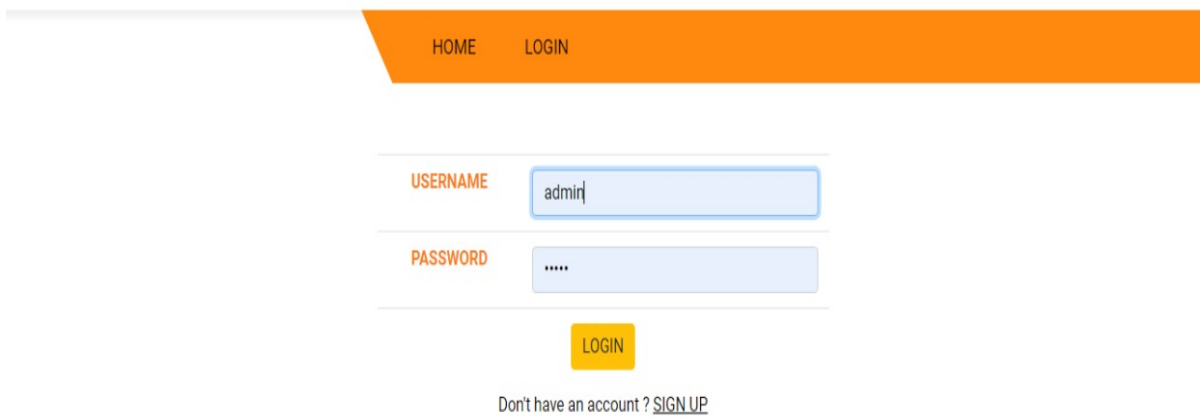


Figure A.5: Admin login page

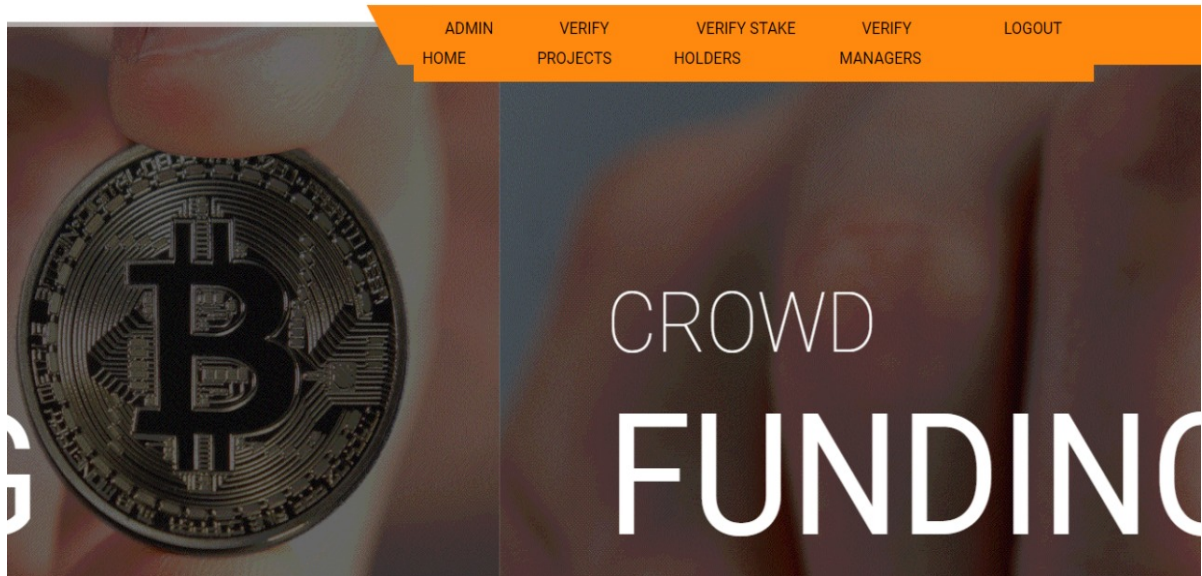


Figure A.6: Admin home page

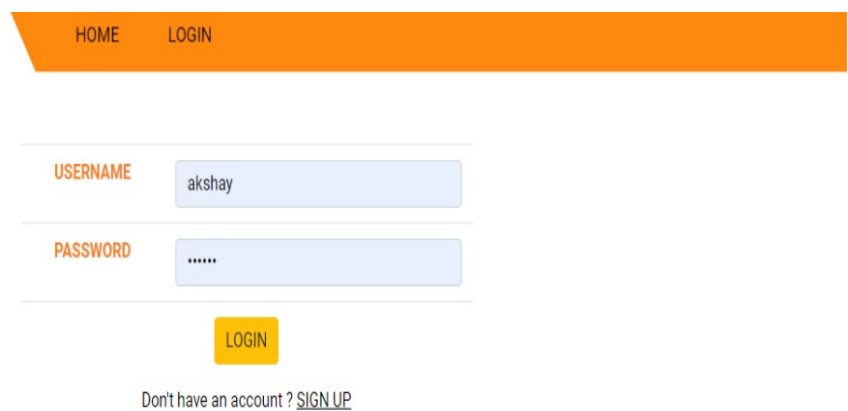


Figure A.7: Manger login page

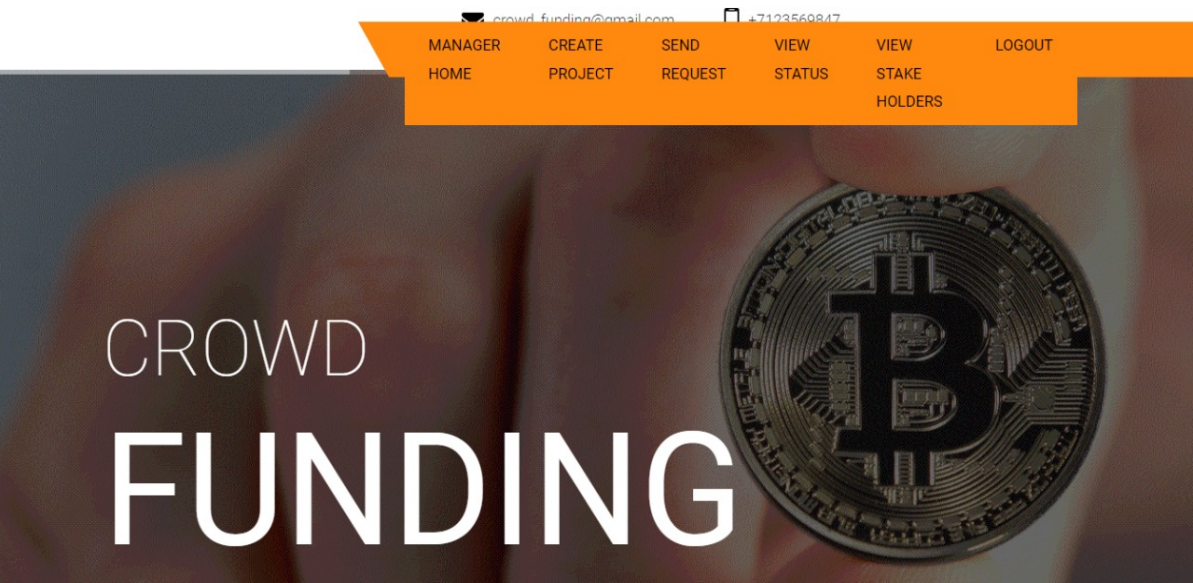


Figure A.8: Manager home page

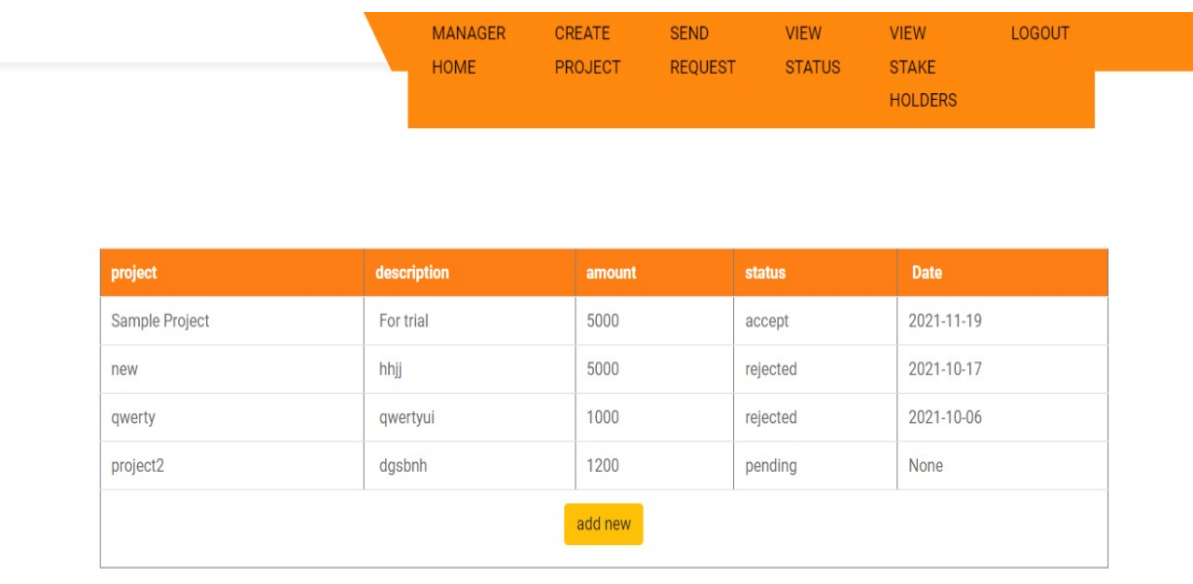


Figure A.9: Create projects

Appendix

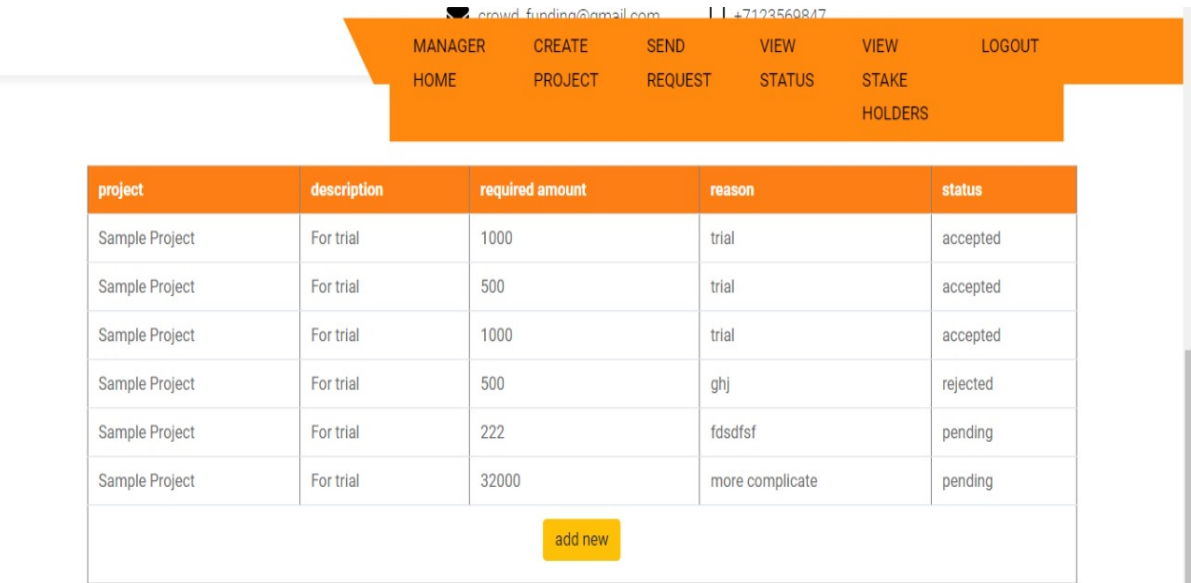


Figure A.10: Sent request

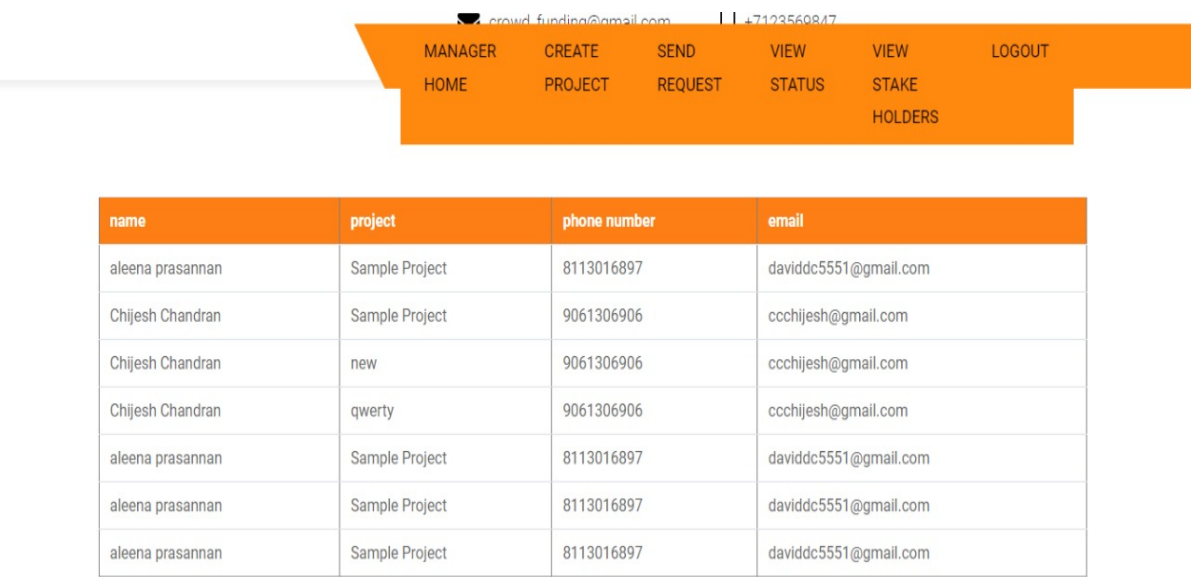


Figure A.11: View Stakeholder

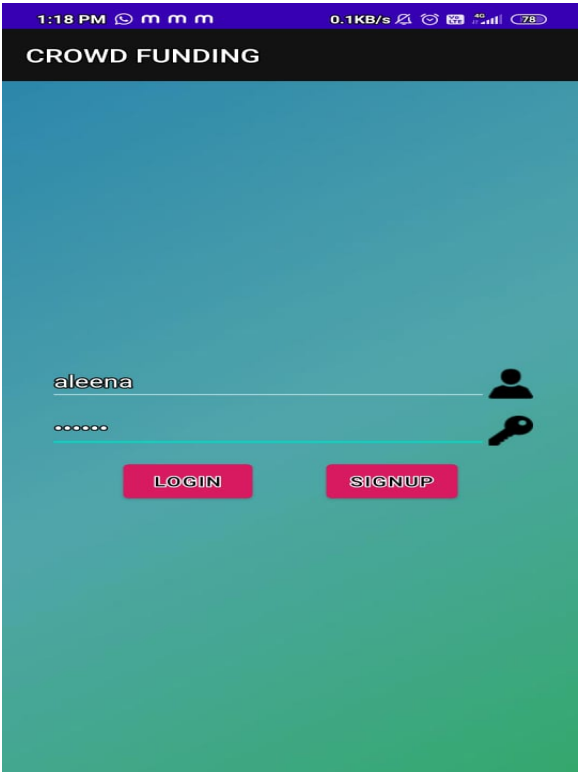


Figure A.12: login stakeholder

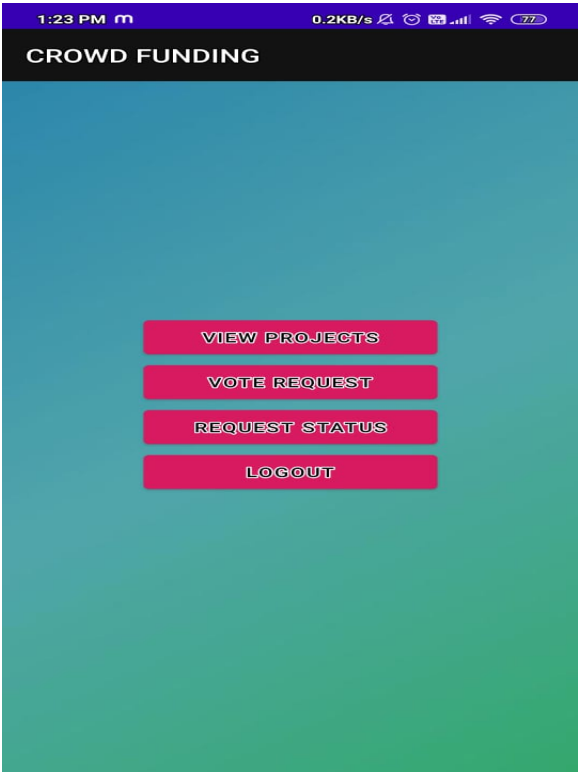


Figure A.13: stakeholder home page

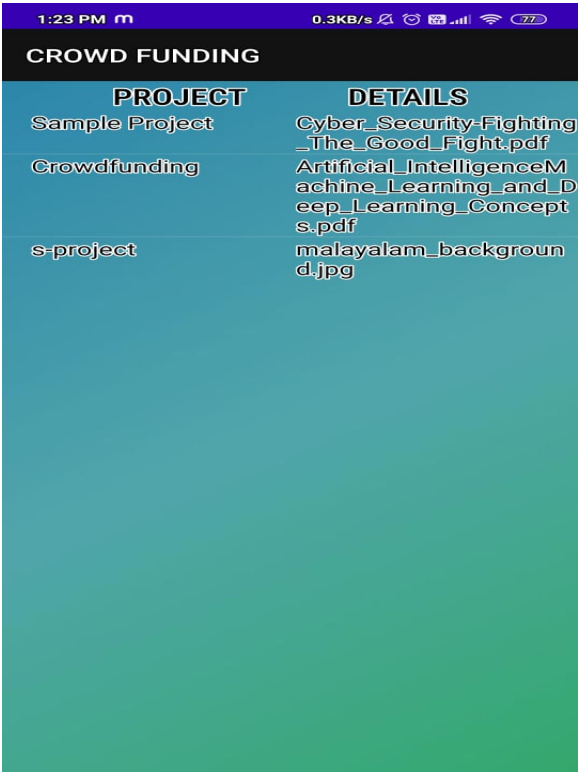


Figure A.14: View projects

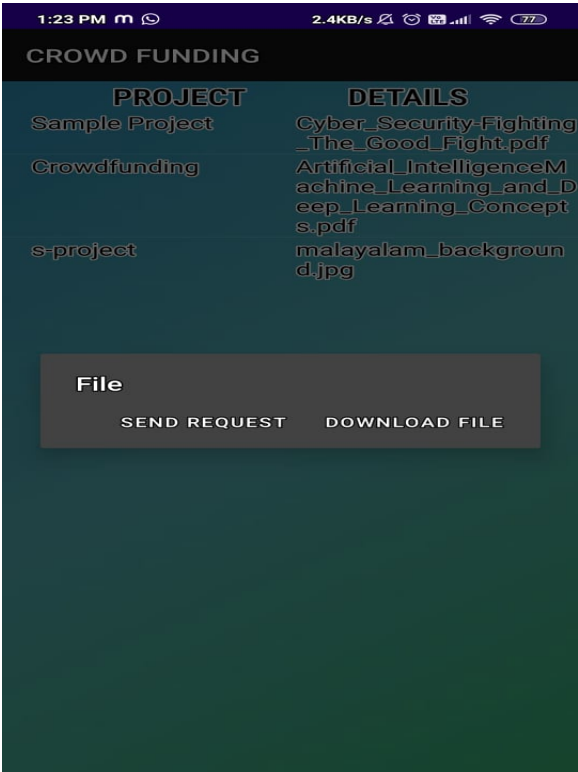


Figure A.15: Vote for request

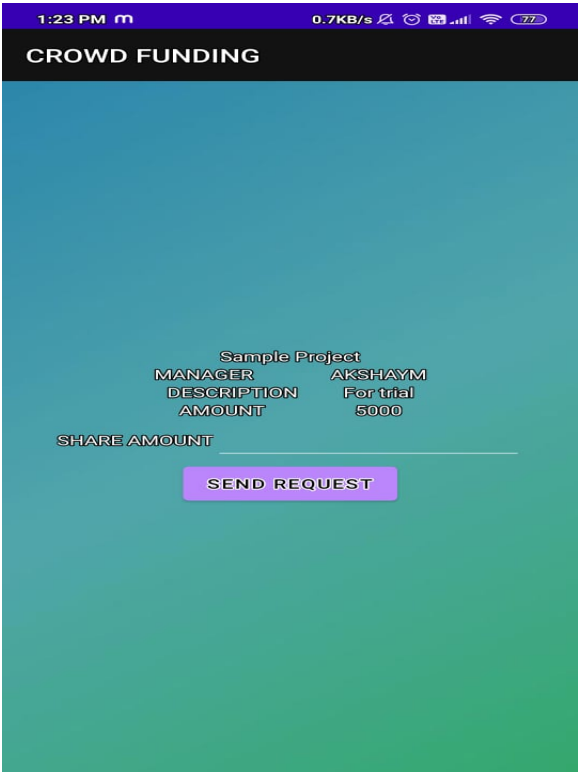


Figure A.16: send request

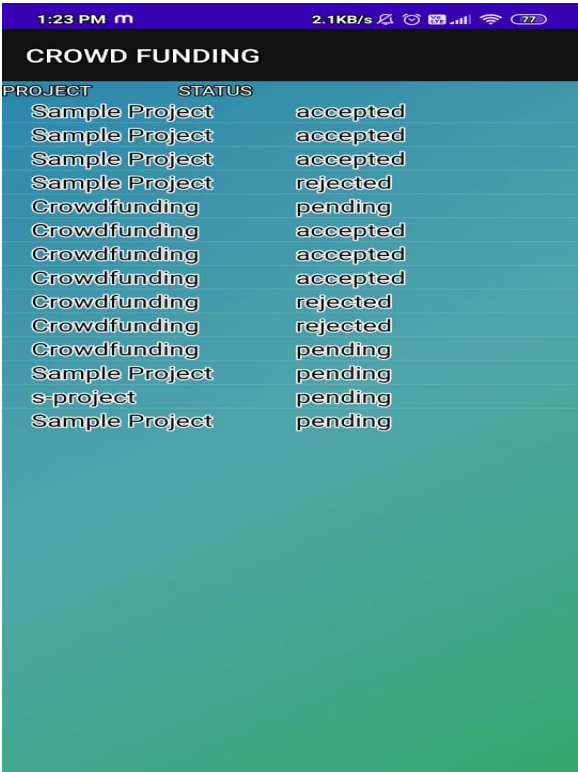



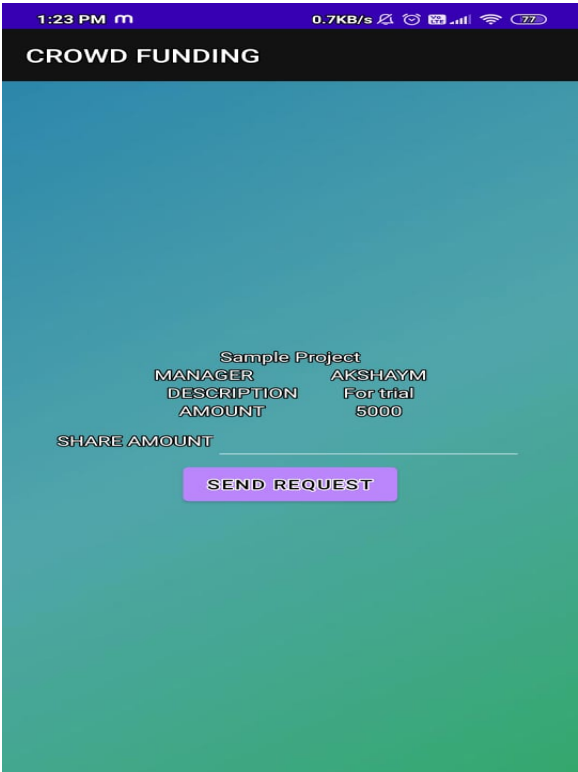
Figure A.17: view request status



A screenshot of a mobile application interface titled "CROWD FUNDING". It displays a list of projects with their names and corresponding amounts. The list includes "Sample Project" and "Crowdfunding" entries with various values. The background is a teal-to-green gradient.

PROJECT	AMOUNT
Sample Project	1000
Sample Project	500
Sample Project	1000
Sample Project	500
Crowdfunding	1000
Crowdfunding	2000
Crowdfunding	1000
Crowdfunding	500
Crowdfunding	100
Crowdfunding	1000
Crowdfunding	333
Sample Project	222
s-project	5000
Sample Project	32000

Figure A.18: view project funt




A screenshot of a mobile application interface titled "CROWD FUNDING". It shows a form for sending a request. The form includes fields for "MANAGER", "DESCRIPTION", and "AMOUNT", with a "SHARE AMOUNT" label. A "SEND REQUEST" button is at the bottom. The background is a teal-to-green gradient.

Sample Project
MANAGER AKSHAYM
DESCRIPTION For trial
AMOUNT 5000

SHARE AMOUNT

SEND REQUEST

Figure A.19: send request



The screenshot shows a mobile application interface with a purple status bar at the top displaying the time 1:23 PM, signal strength, and battery level at 77%. Below the status bar is a black header with the text 'CROWD FUNDING' in white. The main content area has a teal-to-green gradient background and displays a table of accepted projects. The table has two columns: 'PROJECT' and 'AMOUNT'.

PROJECT	AMOUNT
Sample Project	1000
Sample Project	500
Sample Project	1000
Sample Project	500
Crowdfunding	1000
Crowdfunding	2000
Crowdfunding	1000
Crowdfunding	500
Crowdfunding	100
Crowdfunding	1000
Crowdfunding	333
Sample Project	222
s-project	5000
Sample Project	32000

Figure A.20: view accepted projects

4.2 Project Plan

User Story ID	Task Name	Start Date	End Date	Days	Status
1	Sprint 1	31/12/2021	31/11/21	10	Completed
2		31/12/2021	31/11/21		Completed
3		31/12/2021	31/12/2021		completed
4	Sprint 2	11/12/2021	11/12/2021	11	Completed
5		11/12/2021	11/12/2021		Completed
6		11/12/2021	11/12/2021	8	Completed
7		24/12/2021	24/12/2021		Completed
8		2/01/2022	2/01/2022	6	Completed
9		5/01/2022	5/01/2022		Completed

Figure A.21: Project Plan

User Story ID	Task Name	Start Date	End Date	Days	Status
1	Sprint 3	10/01/2022	10/01/2022	10	Completed
2		17/01/2022	17/01/2022		Completed
3		25/01/2023	25/01/2022		completed
4		25/01/2023	25/01/2023	11	completed
5	Sprint 4	1/02/2022	1/02/2022		completed
6		1/02/2022	1/02/2022	8	completed
7		5/02/2022	5/02/2022		completed
8		5/02/2022	5/02/2022	6	completed
9		5/01/2022	5/02/2022		completed

Figure A.22: Project Plan

4.3 Product Backlog

User story ID	Priority <High/Medium/Low>	Size (Hours)	Sprint <#>	Status <Planned/In progress/Completed>	Release Date	Release Goal
1	Medium	2	1	Completed	31/11/21	Table design
2	High	3		Completed	31/11/21	Form design
3	High	5		Completed	31/11/21	Code design
4	Medium	5		Completed	11/12/2021	Project creation and add new project ideas
5	High	5	2	Completed	11/12/2021	View project current status
6	High	5		Completed	11/12/2021	Create request and view sp
7	Medium	5		Completed	24/12/2021	Register for stakeholders by details
8	High	5		Completed	2/01/2022	Login with username and

Figure A.23: Product Backlog

User story ID	Priority <High/Medium/Low>	Size (Hours)	Sprint <#>	Status <Planned/In progress/Completed>	Release Date	Release Goal
9	Medium	2	3	Completed	5/01/2022	View project info from manager
10	High	3		Completed	10/01/2022	Send fund amount to a project
11	High	5		Completed	17/01/2022	View project fund
12	Medium	5		Completed	25/01/2023	View spending request
13	High	5	4	Completed	1/02/2022	Accept/reject spending request
14	High	5		Completed	5/02/2022	test data
15	Medium	5		Completed	5/02/2022	Output generation

Figure A.24: Product backlog

4.4 Sprint Plan

Backlog item	Status & completion date	Original estimate in hours	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7	Day 8	Day 9	Day 10	Day 11	Day 12
User Story#1,2														
Table designing	31/11/21	2	1	1	0	0	0	0	0	0	0	0	0	0
Form designing	31/11/21	5	0	1	1	1	0	0	0	0	0	0	0	1
Code designing	31/11/21	5	0	0	0	0	1	1	1	1	0	0	1	1
User story#3,4														
Project creation and add new project ideas	11/12/2021	5	1	1	0	1	1	1	0	0	1	2	1	1
User story#5,6														
View project current status	24/12/2021	5	1	1	1	0	1	1	0	0	0	1	1	1
Login with username and password	2/01/2022													
User story#7,8														
View project info from manager	5/01/2022	5	1	2	1	0	2	2	1	1	0	1	2	3

Figure A.25: Sprint Plan

Backlog item	Status & completion date	Original estimate in hours	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7	Day 8	Day 9	Day 10	Day 11	Day 12
User Story#9, 10														
View project info from stakeholder	10/01/2022	3	1	1	0	0	0	0	0	0	0	0	0	0
Send fund amount to a project	17/01/2022	5	0	1	1	1	0	0	0	0	1	1	1	
User story#11,12														
View project fund	25/01/2023	5	1	1	0	1	1	1	0	0	1	2	1	3
User story#13,14														
Accept/reject spending request	1/02/2022	5	1	1	1	0	1	1	0	0	1	1	1	2
User story#15														
Testing & output	5/02/2022	5	1	2	1	0	2	2	1	1	0	2	1	4
Total		50	4	5	3	2	4	4	2	2	0	4	5	4

Figure A.26: Sprint Plan

4.5 Sprint Actual

Backlog item	Status & completion date	Original estimate in hours	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7	Day 8	Day 9	Day 10	Day 11	Day 12
User Story#1,2														
Table Designing	31/11/21	2	1	1	0	0	0	0	0	0	0	0	0	0
Form Designing	31/11/21	5	0	1	1	1	0	0	0	0	0	0	0	1
Code Designing	31/11/21	5	0	0	0	0	1	1	1	1	0	0	1	1
User Story#3,4														
Project creation and add new project ideas	11/12/2021	5	1	1	0	1	1	1	0	0	1	2	1	1
User Story#5,6														
View project current status	24/12/2021	5	1	1	1	0	1	1	0	0	0	1	1	1
Login with username and password	2/01/2022													
User Story#7,8														
View project info from manager	5/01/2022	5	1	2	1	0	2	2	1	1	0	1	2	3

Figure A.27: Sprint Actual

Backlog item	Status & completion date	Original estimate in hours	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7	Day 8	Day 9	Day 10	Day 11	Day 12
User Story#9,10														
View project info from stakeholder	10/01/2022	3	1	1	0	0	0	0	0	0	0	0	0	0
Send fund amount to a project	17/01/2022	5	0	1	1	1	0	0	0	0	1	1	1	
User story#11,12														
View project fund	25/01/2023	5	1	1	0	1	1	1	0	0	1	2	1	3
User story#13,14														
Accept/reject spending request	1/02/2022	5	1	1	1	0	1	1	0	0	1	1	1	2
User story#15														
Testing & output	5/02/2022	5	1	2	1	0	2	2	1	1	0	2	1	4
Total		50	4	5	3	2	4	4	2	2	0	4	5	4

Figure A.28: Sprint Actual