A Project Report on

Online Certificate Generation & Verification using Blockchain Framework

Submitted in fulfillment of the requirements for the award
of the degree of

Bachelor of Engineering

in

Information Technology

by

Prasad Jadhav(17104003)
Rutwik Gaikwad(17104074)
Aseem Godambe(17104058)

Under the Guidance of

Prof. Kiran Deshpande

Department of Information Technology
NBA Accredited
A.P. Shah Institute of Technology
G.B.Road,Kasarvadavli, Thane(W), Mumbai-400615
UNIVERSITY OF MUMBAI

Academic Year 2020-2021

Approval Sheet

This Project Report entitled ***"Online Certificate Generation & Verification using Blockchain Framework"*** Submitted by ***"Prasad Jadhav"(17104003), "Rutwik Gaikwad"(17104074), "Aseem Godambe"(17104058)*** is approved for the fulfillment of the requirement for the award of the degree of ***Bachelor of Engineering*** in ***Information Technology*** from ***University of Mumbai***.

Prof. Kiran Deshpande
Guide

Prof. Kiran Deshpande
Head Department of Information Technology

Place:A.P.Shah Institute of Technology, Thane
Date:

CERTIFICATE

This is to certify that the project entitled ***"Online Certificate Generation & Verification using Blockchain Framework"*** submitted by ***"Prasad Jadhav" (17104003), "Rutwik Gaikwad" (17104074), "Aseem Godambe" (17104058)*** for the fulfillment of the requirement for award of a degree ***Bachelor of Engineering*** in ***Information Technology***,to the University of Mumbai,is a bonafide work carried out during academic year 2020-2021.

Prof. Kiran Deshpande
Guide

Prof. Kiran Deshpande
Head Department of Information Technology

Dr. Uttam D.Kolekar
Principal

External Examiner(s)

1.

2.

Place:A.P.Shah Institute of Technology, Thane
Date:

Declaration

We declare that this written submission represents our ideas in our own words and where others' ideas or words have been included, We have adequately cited and referenced the original sources. We also declare that We have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

(Signature)

(Prasad Jadhav 17104003)
(Rutwik Gaikwad 17104074)
(Aseem Godambe 17104058)

Date:

Abstract

In the modern digital age, the advancement in technology has resulted in solving complex problems. But this advancement has also resulted in increased illegal activities. One such major activity is forgery of documents and certificates of an individual. There are many cases reported of certificate forgery everyday and many of them go undetected. The purpose of this study is to develop a system that would authenticate certificates to their real owners and could verify the same on a specific portal. To achieve this, in this project we focus on Ethereum Blockchain as it is immutable, transparent, scalable, and also cost-effective. Universities would be able to generate their own certificates on the web portal itself by uploading their certificate template in Scalable Vector Graphics(SVG) format and the student data in Comma-separated Values(CSV) format. Each generated certificate would be allotted to the respective student within the system. Each certificate would be given a unique hash code so that it could be verified easily and there would be no scope of duplication. The authentication and verification would be done on the same web portal. To sum up, this method would save time and efforts that are required to verify a certificate manually and would result in an effective, secure way to generate certificates.

Contents

List of Figures

List of Tables

List of Abbreviations

SVG:	Support Vector Graphics
CSV:	Comma-separated values
QR code:	Quick Response code
OTP:	One Time Password
IPFS:	Interplanetary File System
HTTP:	Hypertext Transfer Protocol
URL:	Uniform Resource Locator
EVM:	Ethereum Virtual Machine

Chapter 1

Introduction

Everyone has a particular talent or excels in a field and certificates are a great way to reflect the achievements. It proves that an individual has acquired knowledge of a particular skill. Companies need skilled employees and certificates prove to be a valid solution to depict once capabilities. Displaying a certificate of a particular skill verifies that the certificate holder is competent in that skill and helps the company in the hiring process. Also due to the current pandemic situation, online learning and certifications through online courses has drastically increased. People have started to adapt to online learning and hence the certificates hold a greater value to validate the progress of an individual and also it adds a great value to their profiles.

As certificates hold such higher values people tend to misuse it and generate a fake copy of the certificate of the skill which they have not acquired. The companies or the organisation where an individual is applying offers an advantage to the person having the necessary certificates. It's difficult for the organisation to check individual certificates as they may have to contact the issuer and it takes a lot of time and effort. The fake certificates can be easily generated using any online website or normal photo editing software. As a result it makes it more difficult for the deserving candidate to have all the genuine certificates.A survey conducted by UK's national qualifications agency UK NARIC conducted across 17 countries in different universities found the common problem of difficulty in verifying documents. According to the survey, it has been found that 62% of them verify the documents by contacting the institutes awarding them while 14% of them didn't even bother to check the originality of the certificate. It stated that around 75% of the certificates submitted were fraud [1]. Similar research and investigation conducted by the BBC Radio 4's File on Four programme found out about thousands of fake degree certificates in the UK from "Diploma Mill" in Pakistan. This deteriorates the standard of employees being hired and unfair against the students who are hardworking and achieving the results [2].

The motive of this research is to create a system which is secure, easy to access and verify certificates. The organisation hiring an individual could verify the certificate online without the need to contact the issuer. Blockchain technology along with different hashing methods is used to maintain a secure record of the information. An SVG(Scalable Vector Graphics) format certificate template is used specifically to reduce the cost of storing multiple images on the blockchain network. This creates a scalable and economical system for an organisation to implement.

1.1 Scope

Manually generating and verifying a certificate takes a huge amount of time for any institution or an University. So due to this issue, having a single portal for certificate generation and verification will save a lot of time of both the Institution/University as well as the student.

1.2 Objectives

1. **Central Portal for all**: This system makes it possible for all the Universities/ Institutions to generate their certificates on one single web portal with their own private educational email Ids.

2. **Throughout data security**: Ethereum blockchain makes this system highly secure as the data cannot be overwritten and the certificate cannot be generated without the admin's permission.This system would also take care of a certificate's security as each certificate generated has a unique hash code making sure that there is no scope of duplicacy.

3. **Certificate generation**: Universities would be able to generate their own certificates on the web portal itself. Universities/Institutions will have to upload their certificate template in Scalable Vector Graphics(SVG) format and the student data in Comma-separated Values(CSV) format. Each generated certificate would be allotted to the respective student within the system.

4. **Easy to use**: It is ensured that the web portal is easily accessible and very easy to use. The web portal would have a very simple-to-understand user interface making sure that anyone could easily use it without any problem.

5. **Cost effective**: Ethereum Blockchain makes this system highly scalable. But Ethereum cost is a major drawback. To overcome this, the system would not upload the entire certificate on the blockchain and upload only the mapping data(data that is to be mapped on the certificate), since this mapping data is very less the cost of the Ethereum would be very low and affordable, thus making it cost effective.

Chapter 2

Literature Review

2.0.1 Shanmuga Priya R,Swetha N 'Online Certificate Validation Using Blockchain.' [6]

This document summarizes the problems of forged certificates and how blockchain can solve the issue. Netbeans IDE and Android Studio are used to develop the server communication and the application to scan the QR codes. EthereumJS is used for faster Ethereum applications. The application involves the user uploading his/her certificate like 10th-grade mark sheet, college certificates, government certificates, and so on to the portal. After uploading the certificate, the data is then sent to the issuer for validation. The issuer (example: School which is responsible for validation of 10th grade Marksheet) has to validate the data received by the application for verification. Once verification is successful, the data will be stored on the server else it would be discarded. On the mobile application, a QR code will be generated based on the certificate number. The QR code can then be shared with anyone else for verification in case of necessity. When the QR code is scanned, an OTP (One Time Password) will be sent to the registered mobile number for verification. After proper authentication, the user can view the certificate. If the number of scans goes beyond the permitted limit, the location of the scanner will be sent to the authorized user with a permission link. From that link, the authorized user can either allow or deny the person. The major drawback of this system is that verification can be delayed by the issuer as the certificate has to reach the organization for verification and the data is sent to blockchain only after verification. The OTP system can get tedious. This system can get costly as well.

2.0.2 Nitin Kumavat, Swapnil Mengade, Dishant Desai, Jesal-Varolia 'Certificate Verification System using Blockchain' [7]

This document summarizes that during the course of education the students achieve many certificates. Students produce these certificates while applying for jobs in public or private sectors, where all these certificates are needed to be verified manually. There can be incidents where students may produce fake certificates and it is difficult to identify them. The solution proposed in this system uses Ethereum and IPFS (Interplanetary File System). IPFS is a peer to peer, content address system. It is very similar to BitTorrent and MerkleDag. Unlike HTTP which restricts or provides low latency on transfer of large amounts over the network which uses IP addressing, IPFS uses content addressing. As a result it creates a distributed system of different nodes across the network. It returns us a hash value and it uses this value to retrieve the data. They propose to add the data with the certificate on the IPFS network to generate the hash for it and later on add the data to the blockchain network with the help of EVM. This topology includes two distributed networks to provide additional security to the model. While retrieving the certificate they would compare the hash and pull the entire certificate which has been stored on the IPFS network. The advantage of this system was it is more secure, provides reliability. With two distributed networks, it's almost impossible to tamper the data. On the other hand, the disadvantage of the system is that IPFS is a tedious process to set-up as well as storing the certificate image on any distributed blockchain platform costs a lot of money and hinders the scalability of the system.

Chapter 3

Proposed System Architecture

3.0.1 Methodology

This proposed system revolves around creating a secure and fast method to generate and verify the certificates using the features of SVG and blockchain. The certificate's data is hashed and stored on a blockchain network to provide security and immutability to the data. On the blockchain network, this data is stored on blocks. The block contains the hashed data, timestamp, and the id of the next block. Then the block is added to the network. The template of the certificate which is in SVG format is stored on a database server that is not a blockchain-based network as the certificate template won't be modified. The data which is hashed includes the name of the certificate holder, their email, issuing date, expiry date for the certificate, and information regarding other fields of the certificate. This data can only be submitted by the certificate issuer and thus ensuring the right data is provided. Anyone can check certificates of the students using the unique id provided to each student for their certificates or from a student's profile.

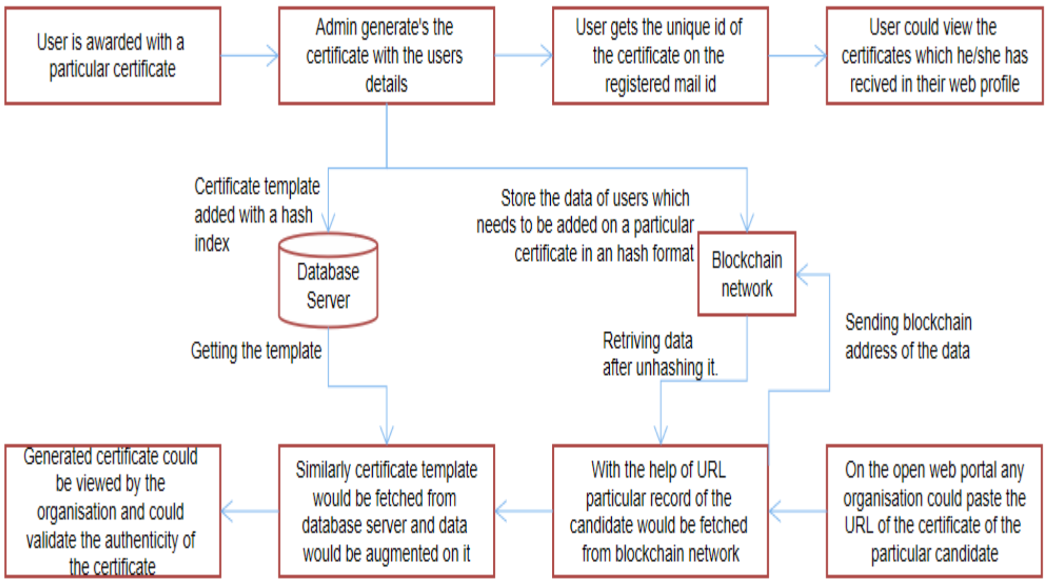
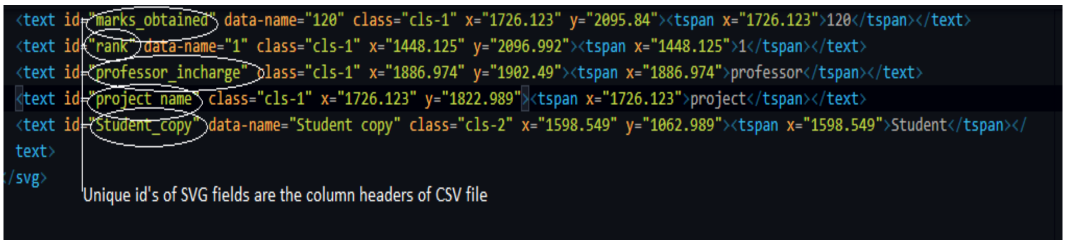

User is awarded with a particular certificate
Admin generate's the certificate with the users details
User gets the unique id of the certificate on the registered mail id
User could view the certificates which he/she has recived in their web profile
Certificate template added with a hash index
Database Server
Store the data of users which needs to be added on a particular certificate in an hash format
Blockchain network
Getting the template
Retriving data after unhashing it.
Sending blockchain address of the data
Generated certificate could be viewed by the organisation and could validate the authenticity of the certificate
Similarly certificate template would be fetched from database server and data would be augmented on it
With the help of URL particular record of the candidate would be fetched from blockchain network
On the open web portal any organisation could paste the URL of the certificate of the particular candidate

Figure 3.1: Architecture of proposed system.

3.0.2 Certificate Generation

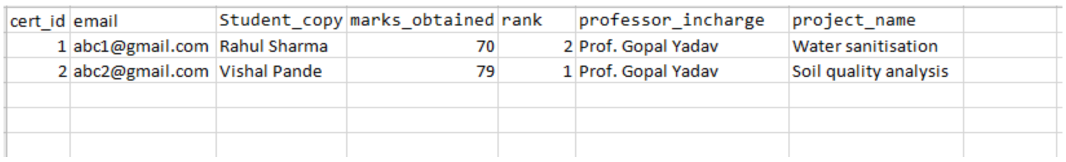
Certificates can be generated only by the organizations registered on the website. They need to upload the template of the certificate in the SVG format. The SVG template should have a proper id for a particular field like the space for the name of the receiver should have a well-defined id, for example, 's_name'. Once the SVG template is uploaded the organization needs to upload a CSV file that has the record of the students who have won the certificates. The CSV needs to have the first row as the ids of the fields marked on the SVG template, along with it the CSV needs to have a compulsory email column that would contain the email of the receiver. Other columns of the CSV would be the remaining values that needed to be filled on the template like the receiver's name, issuing date and expiry date of the certificate, etc. The only condition is that the id of the field written on the SVG template should be the column header of the CSV file. This method is done to ensure that the blanks on the template could be filled by the right data in the CSV file. Once the upload action of the template and CSV file is done the certificates would be auto-generated for all the students receiving the certificates and would be available in their profile on the website. This would be done with the help of email ids associated with each student present in the CSV file.



```
<text id="marks_obtained" data-name="120" class="cls-1" x="1726.123" y="2095.84"><tspan x="1726.123">120</tspan></text>
<text id="rank" data-name="1" class="cls-1" x="1448.125" y="2096.992"><tspan x="1448.125">1</tspan></text>
<text id="professor_incharge" class="cls-1" x="1886.974" y="1902.49"><tspan x="1886.974">professor</tspan></text>
<text id="project_name" class="cls-1" x="1726.123" y="1822.989"><tspan x="1726.123">project</tspan></text>
<text id="Student_copy" data-name="Student copy" class="cls-2" x="1598.549" y="1062.989"><tspan x="1598.549">Student</tspan></
text>
</svg>
```

Unique id's of SVG fields are the column headers of CSV file

Figure 3.2: Example of a certificate template in an SVG format indicating the ID's of the fields of the certificate.

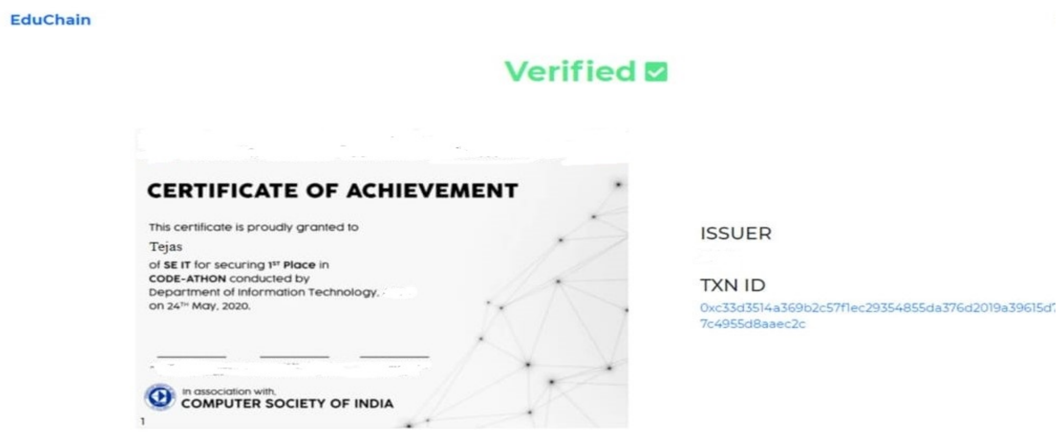


cert_id	email	Student_copy	marks_obtained	rank	professor_incharge	project_name
1	abc1@gmail.com	Rahul Sharma	70	2	Prof. Gopal Yadav	Water sanitisation
2	abc2@gmail.com	Vishal Pande	79	1	Prof. Gopal Yadav	Soil quality analysis

Figure 3.3: CVS file with the data of students winning the certificate for the above given SVG template.

3.0.3 Certificate Validation

In this process, the certificates are validated. As the data is stored on the blockchain network there is no way to edit the data once it's appended into the blocks on the network. The process works as when the certificate data is uploaded by the organization in CSV format the data is extracted from the CSV and uploaded on the blockchain network. Once the data is uploaded on the network, students who have won those certificates are informed through an email that they can view their certificates now in their profile and are provided a unique key or URL for the certificate. Students could use or provide this key, URL to anyone to authenticate the certificates. When a particular key is used the data of that candidate is retrieved from the blockchain and the corresponding template is fetched from the database server. The data from the blockchain is mapped on the SVG templates as the fields of the template are pre-recorded and thus generates a certificate in front of the user which is not tampered with by any means. Thus an entity could verify that the certificate is genuine and possessed by the actual owner of the certificate.


EduChain

Verified ✓

CERTIFICATE OF ACHIEVEMENT

This certificate is proudly granted to

Tejas

of **SE IT** for securing 1ST **Place** in
CODE-ATHON conducted by
Department of Information Technology,
on 24TH May, 2020.

In association with,
COMPUTER SOCIETY OF INDIA

1

ISSUER

TXN ID

0xc33d3514a369b2c57f1ec29354855da376d2019a39615d7
7c4955d8aaec2c

Figure 3.4: Certificate uploaded on the web portal which is verified and genuine certificate.

3.0.4 Working of Application

In our application on the landing page any individual could see a certificate by just entering its unique id on the search bar. There is an admin login, once logged in the admin one could add certificate and CSV file to generate certificates. Once the data is uploaded by the admin it would be stored temporarily in a database and would be uploaded on the blockchain network periodically using a CRON script every day at midnight. This is done to ensure every certificate's data is uploaded on the blockchain network because if we fail to upload any data we cannot rewrite the blocks on the network. Once the data is uploaded students are informed that their certificate is uploaded. When logged in as students they could see the certificates achieved by them in their profile and would see a green tick indicating that the certificate is validated. Students can share their certificate links with anyone to view their certificates.

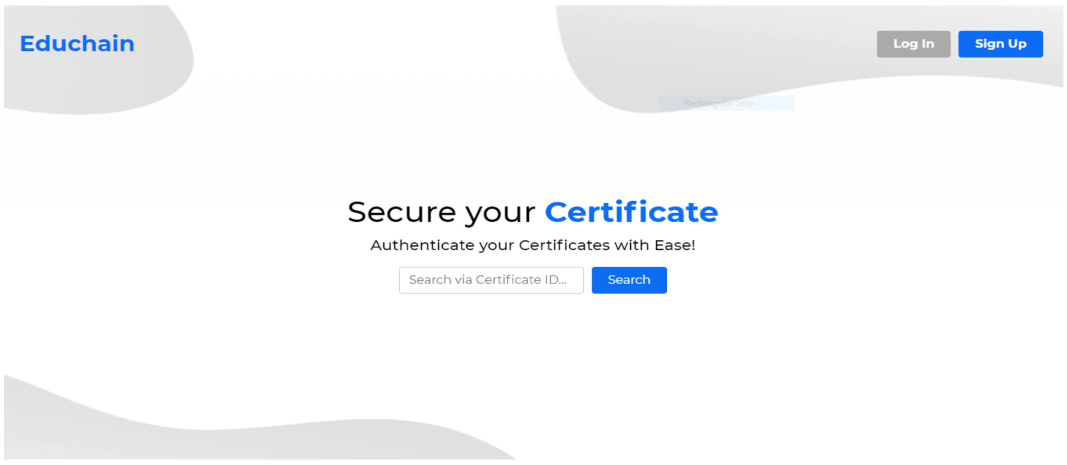
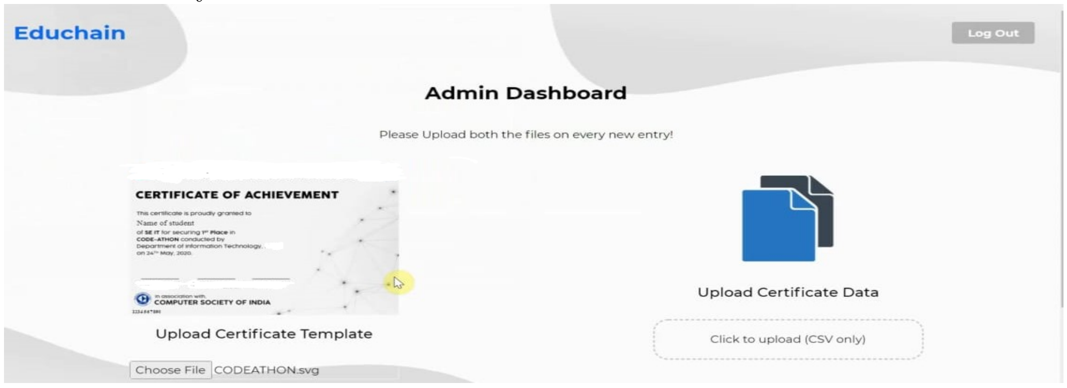

Educhain
Log In
Sign Up
Secure your Certificate
Authenticate your Certificates with Ease!
Search via Certificate ID...
Search

Figure 3.5: Homepage of the web portal where anyone could search for any certificate to verify it's authenticity.


Educhain
Log Out
Admin Dashboard
Please Upload both the files on every new entry!
Upload Certificate Template
Choose File
CODEATHON.svg
Upload Certificate Data
Click to upload (CSV only)

CERTIFICATE OF ACHIEVEMENT

This certificate is proudly granted to
Name of student
of SE IT for securing 1st Place in
CODE-ATHON conducted by
Department of Information Technology,
on 24th May, 2020.

in association with
COMPUTER SOCIETY OF INDIA

Figure 3.6: Admin dashboard to upload SVG and CSV file

Chapter 4

Design

4.0.1 Use-case Diagram

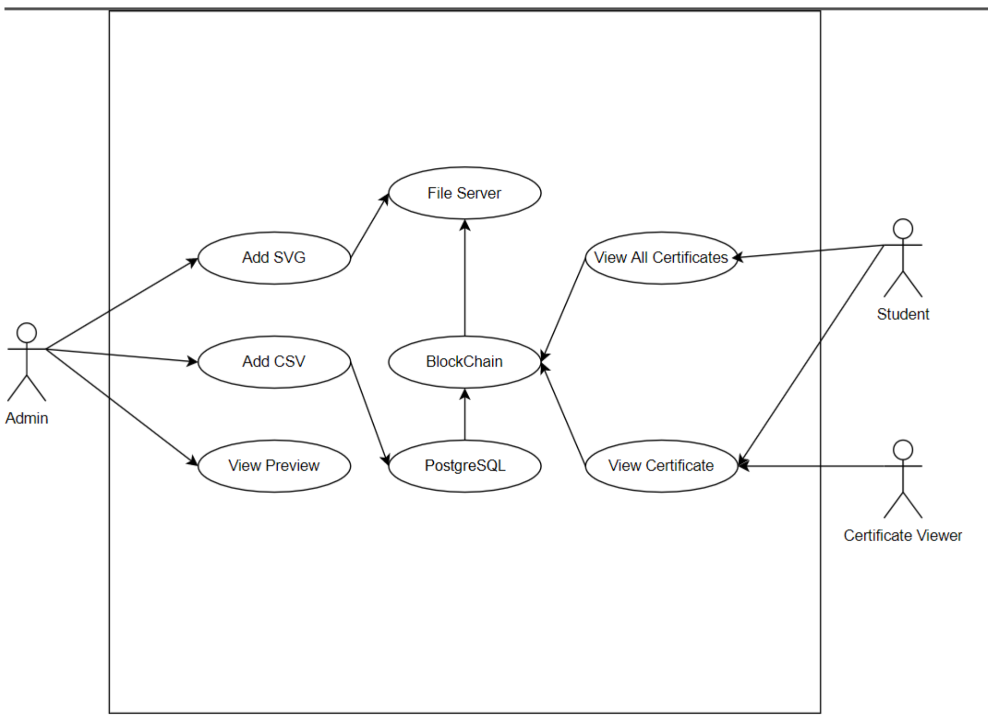
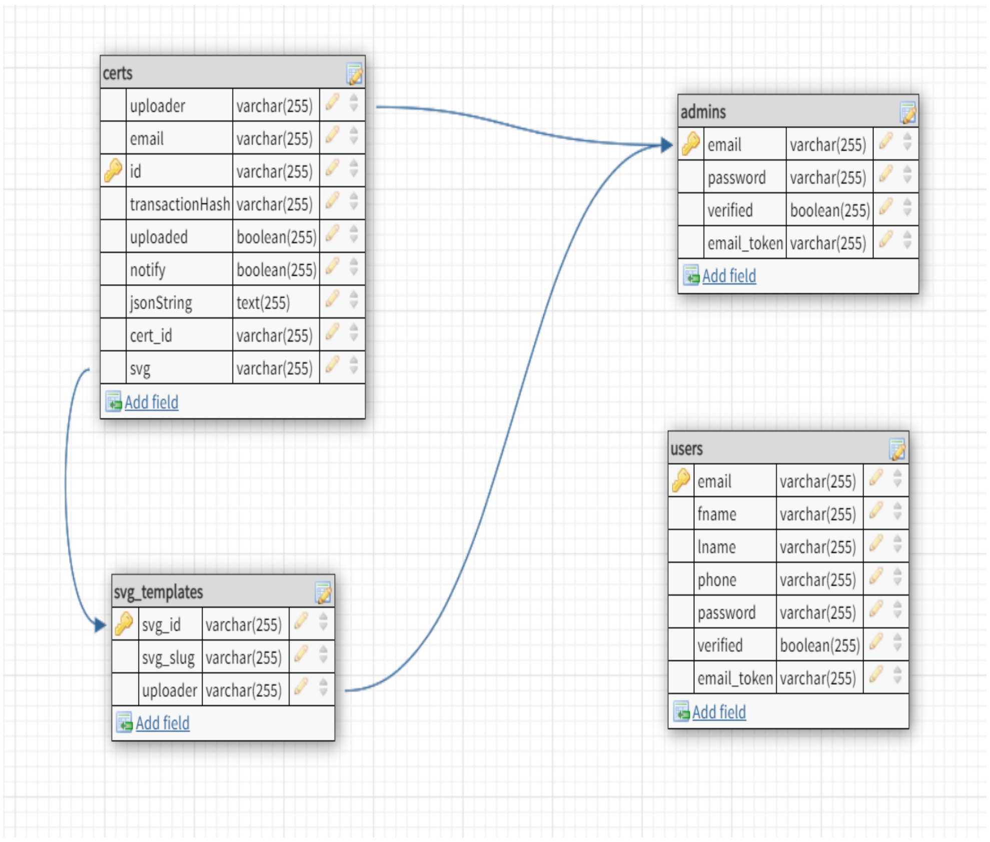


Figure 4.1: Use-case of the system.

In the above figure admin, students and certificate viewers are actors. In case of admin he/she can add SVG template for the certificate, add CSV file and can view the preview of the certificates. The students can interact with the system by viewing the certificates. Anyone else who has the certificate id can also view any particular certificate who acts as a viewer.

4.0.2 Database Schema Diagram



certs	
uploader	varchar(255)
email	varchar(255)
id	varchar(255)
transactionHash	varchar(255)
uploaded	boolean(255)
notify	boolean(255)
jsonString	text(255)
cert_id	varchar(255)
svg	varchar(255)

admins	
email	varchar(255)
password	varchar(255)
verified	boolean(255)
email_token	varchar(255)

users	
email	varchar(255)
fname	varchar(255)
lname	varchar(255)
phone	varchar(255)
password	varchar(255)
verified	boolean(255)
email_token	varchar(255)

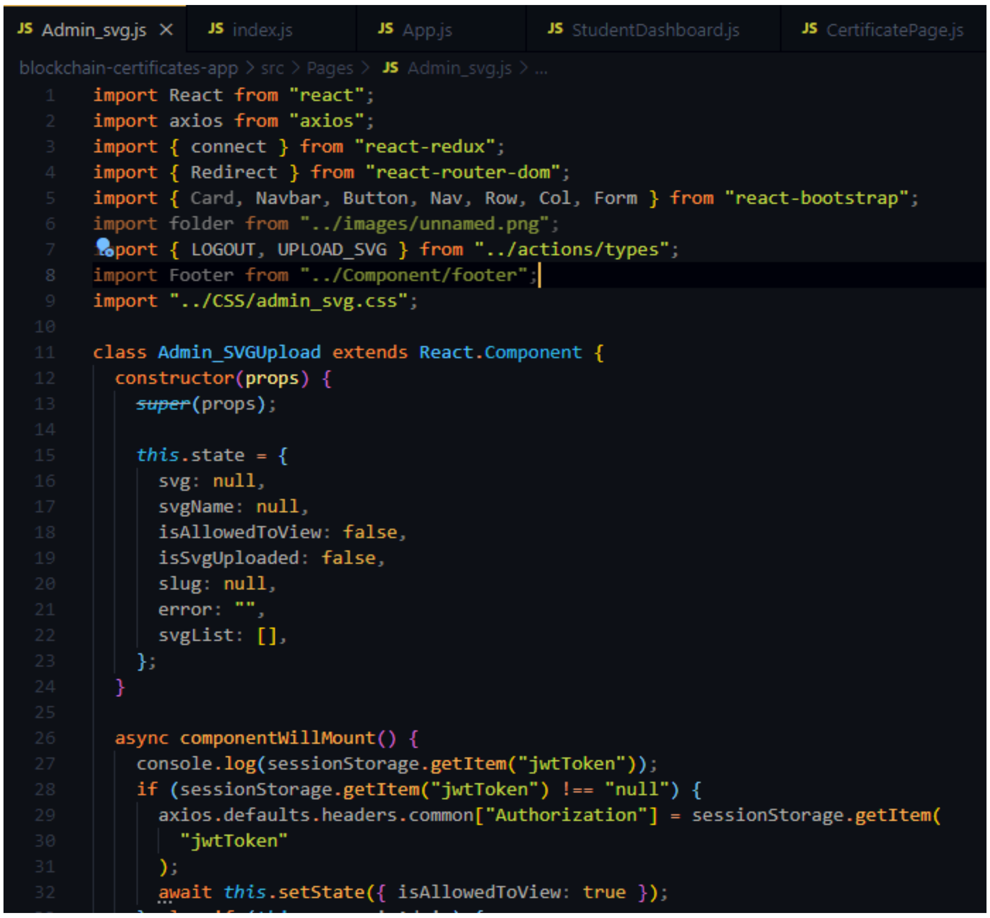
svg_templates	
svg_id	varchar(255)
svg_slug	varchar(255)
uploader	varchar(255)

Figure 4.2: Database Schema diagram of the system.

The above figure depicts the database schema of the system and the relation between different entities in the system.

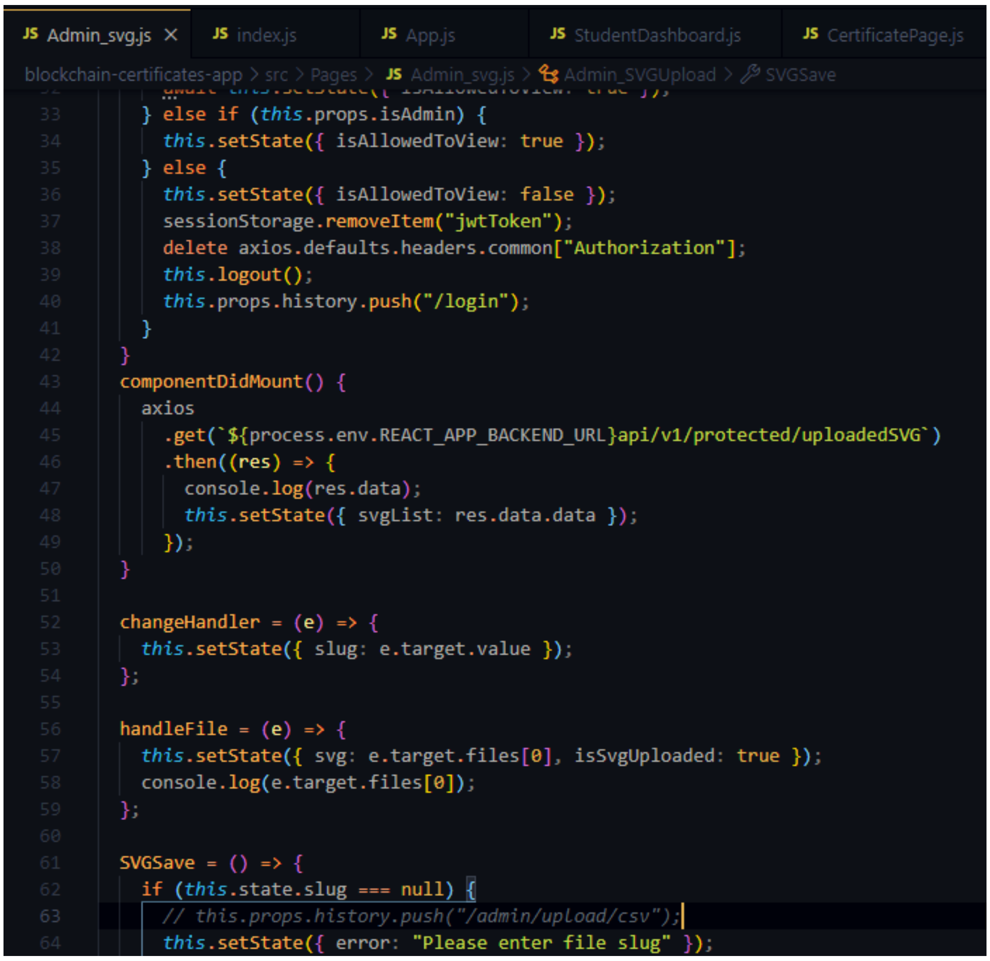
Chapter 5

Implementation



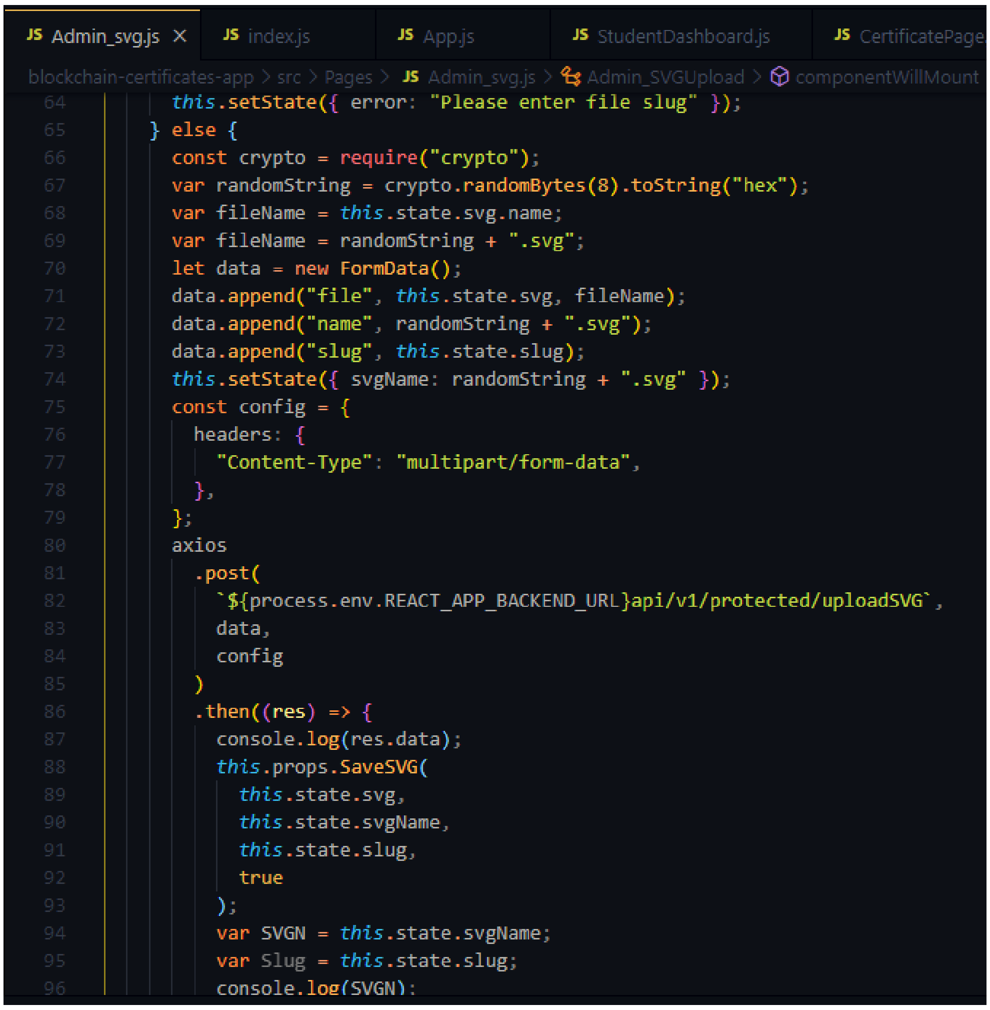
```
import React from "react";
import axios from "axios";
import { connect } from "react-redux";
import { Redirect } from "react-router-dom";
import { Card, Navbar, Button, Nav, Row, Col, Form } from "react-bootstrap";
import folder from "../images/unnamed.png";
import { LOGOUT, UPLOAD_SVG } from "../actions/types";
import Footer from "../Component/footer";
import "../CSS/admin_svg.css";

class Admin_SVGUpload extends React.Component {
  constructor(props) {
    super(props);

    this.state = {
      svg: null,
      svgName: null,
      isAllowedToView: false,
      isSvgUploaded: false,
      slug: null,
      error: "",
      svgList: [],
    };
  }

  async componentWillMount() {
    console.log(sessionStorage.getItem("jwtToken"));
    if (sessionStorage.getItem("jwtToken") !== "null") {
      axios.defaults.headers.common["Authorization"] = sessionStorage.getItem(
        "jwtToken"
      );
      await this.setState({ isAllowedToView: true });
```

Figure 5.1: Program for admin to add SVG template file (a). Importing all the essential packages to process the SVG file uploaded as template. The constructor initializes the state variables. The componentWillMount function runs before the component is about to be mounted. It checks if the jwt token is set or not.


Admin_svg.js
index.js
App.js
StudentDashboard.js
CertificatePage.js
blockchain-certificates-app > src > Pages > Admin_svg.js > Admin_SVGUpload > SVGSave

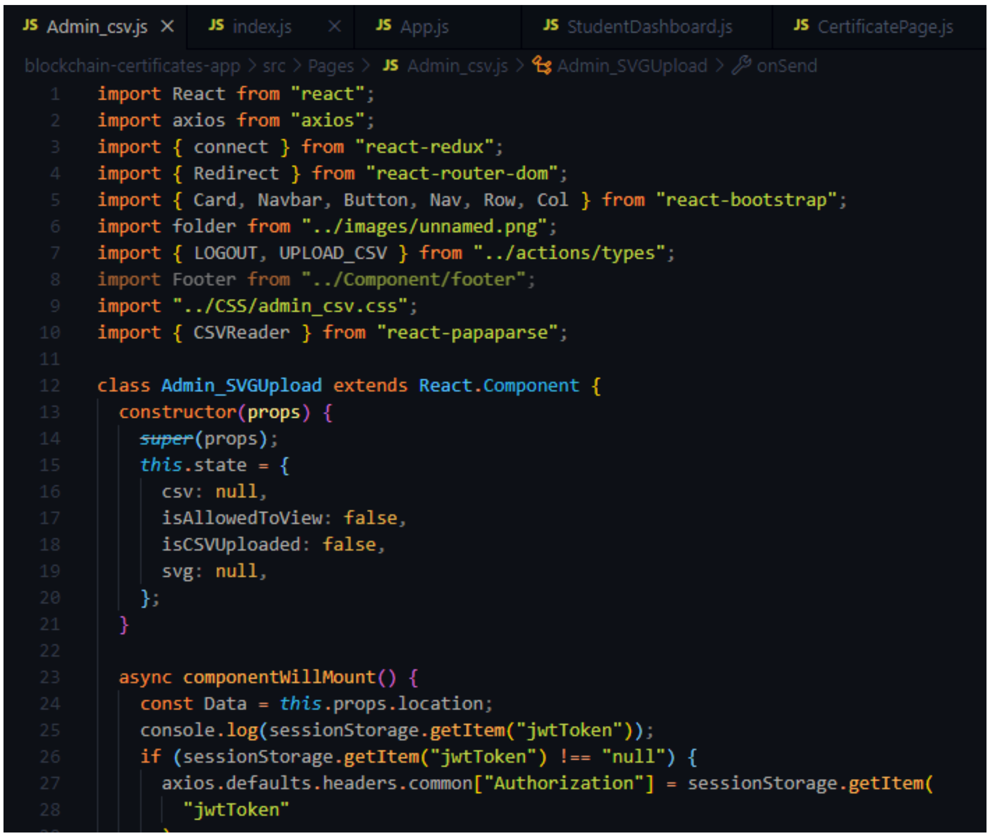
```
      await this.setState({ isAllowedToView: true });
    } else if (this.props.isAdmin) {
      this.setState({ isAllowedToView: true });
    } else {
      this.setState({ isAllowedToView: false });
      sessionStorage.removeItem("jwtToken");
      delete axios.defaults.headers.common["Authorization"];
      this.logout();
      this.props.history.push("/login");
    }
  }
  componentDidMount() {
    axios
      .get(`${process.env.REACT_APP_BACKEND_URL}api/v1/protected/uploadedSVG`)
      .then((res) => {
        console.log(res.data);
        this.setState({ svgList: res.data.data });
      });
  }

  changeHandler = (e) => {
    this.setState({ slug: e.target.value });
  };

  handleFile = (e) => {
    this.setState({ svg: e.target.files[0], isSvgUploaded: true });
    console.log(e.target.files[0]);
  };

  SVGSave = () => {
    if (this.state.slug === null) {
      // this.props.history.push("/admin/upload/csv");
      this.setState({ error: "Please enter file slug" });
```

Figure 5.2: Program for admin to add SVG template file (b). Check for access is done here. The state variables are checked for proper access to the Admin panels UploadSVG page. If any unauthorized login attempt is made, the user will be redirected to the Login Page. The componentDidMount runs after the has been completely mounted. The function makes an API call to upload the SVG file.


Admin_svg.js
index.js
App.js
StudentDashboard.js
CertificatePage
blockchain-certificates-app > src > Pages > Admin_svg.js > Admin_SVGUpload > componentWillMount

```
      this.setState({ error: "Please enter file slug" });
    } else {
      const crypto = require("crypto");
      var randomString = crypto.randomBytes(8).toString("hex");
      var fileName = this.state.svg.name;
      var fileName = randomString + ".svg";
      let data = new FormData();
      data.append("file", this.state.svg, fileName);
      data.append("name", randomString + ".svg");
      data.append("slug", this.state.slug);
      this.setState({ svgName: randomString + ".svg" });
      const config = {
        headers: {
          "Content-Type": "multipart/form-data",
        },
      };
      axios
        .post(
          `${process.env.REACT_APP_BACKEND_URL}api/v1/protected/uploadSVG`,
          data,
          config
        )
        .then((res) => {
          console.log(res.data);
          this.props.SaveSVG(
            this.state.svg,
            this.state.svgName,
            this.state.slug,
            true
          );
          var SVGN = this.state.svgName;
          var Slug = this.state.slug;
          console.log(SVGN);
```

Figure 5.3: Program for admin to add SVG template file (c). The function does the file-Handeling part. It checks if proper file name and slugs are provided by the admin. The configurations are checked and headers are set. The state variables are set for the upload to be done successfully.

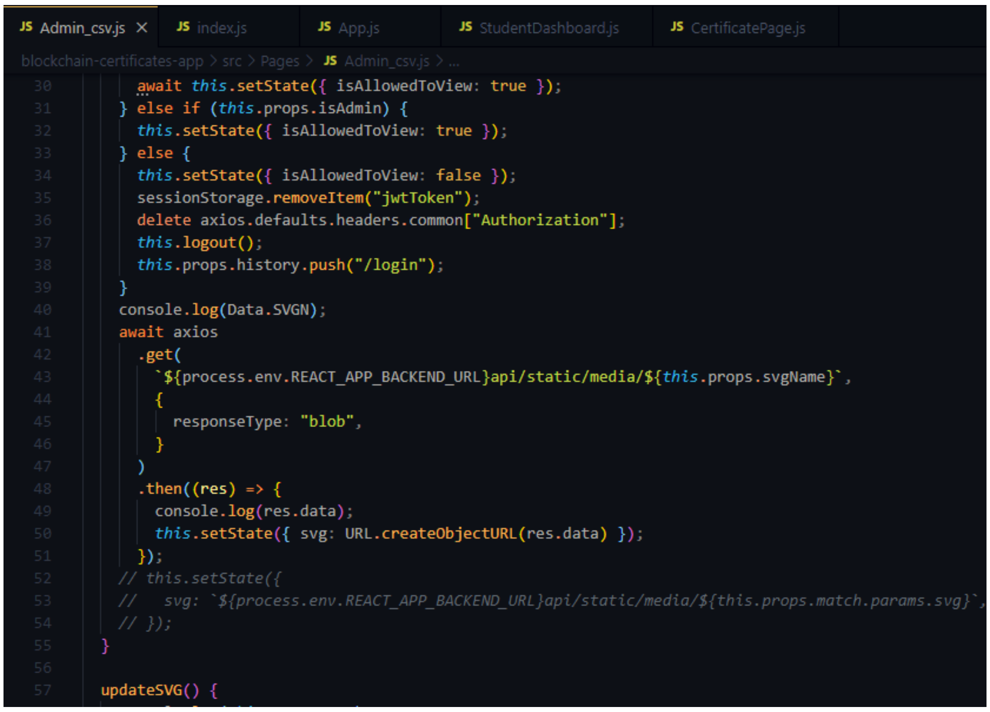


JS Admin_csv.js × JS index.js × JS App.js JS StudentDashboard.js JS CertificatePage.js

blockchain-certificates-app > src > Pages > JS Admin_csv.js > Admin_SVGUpload > onSend

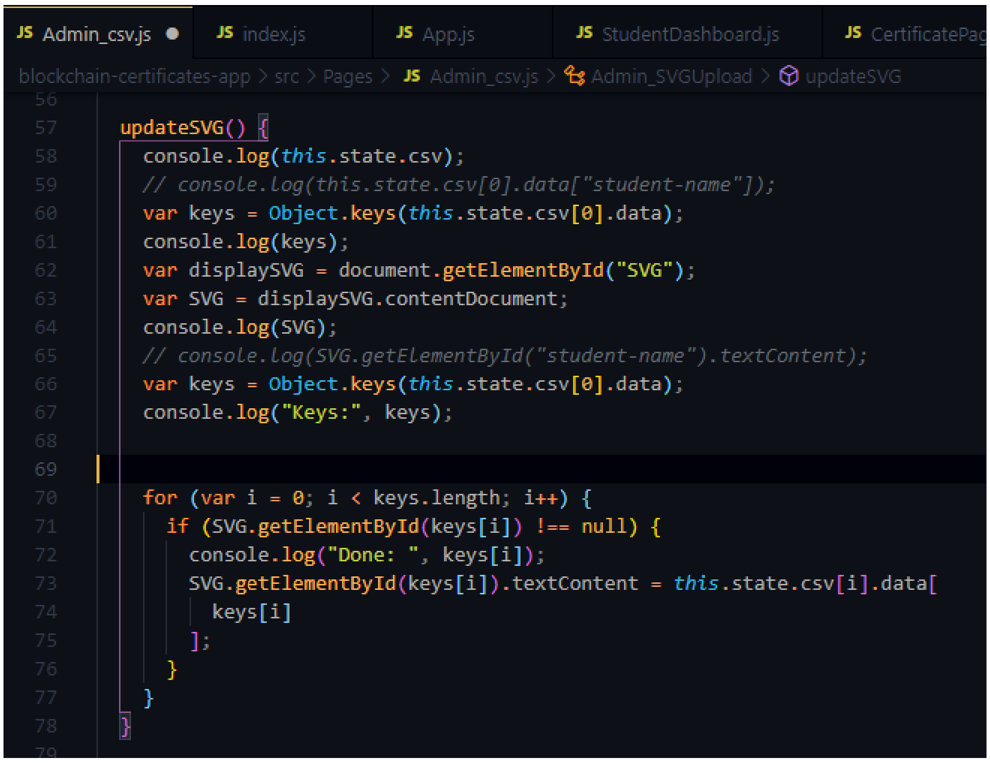
```
import React from "react";
import axios from "axios";
import { connect } from "react-redux";
import { Redirect } from "react-router-dom";
import { Card, Navbar, Button, Nav, Row, Col } from "react-bootstrap";
import folder from "../images/unnamed.png";
import { LOGOUT, UPLOAD_CSV } from "../actions/types";
import Footer from "../Component/footer";
import "../CSS/admin_csv.css";
import { CSVReader } from "react-papaparse";

class Admin_SVGUpload extends React.Component {
  constructor(props) {
    ~~super~~(props);
    this.state = {
      csv: null,
      isAllowedToView: false,
      isCSVUploaded: false,
      svg: null,
    };
  }

  async componentWillMount() {
    const Data = this.props.location;
    console.log(sessionStorage.getItem("jwtToken"));
    if (sessionStorage.getItem("jwtToken") !== "null") {
      axios.defaults.headers.common["Authorization"] = sessionStorage.getItem(
        "jwtToken"
```

Figure 5.4: Program for admin to add CSV file (a). The necessary libraries are imported to process the CSV file containing the student data. The constructor initializes the state variables to hold the csv data after upload done by admin.



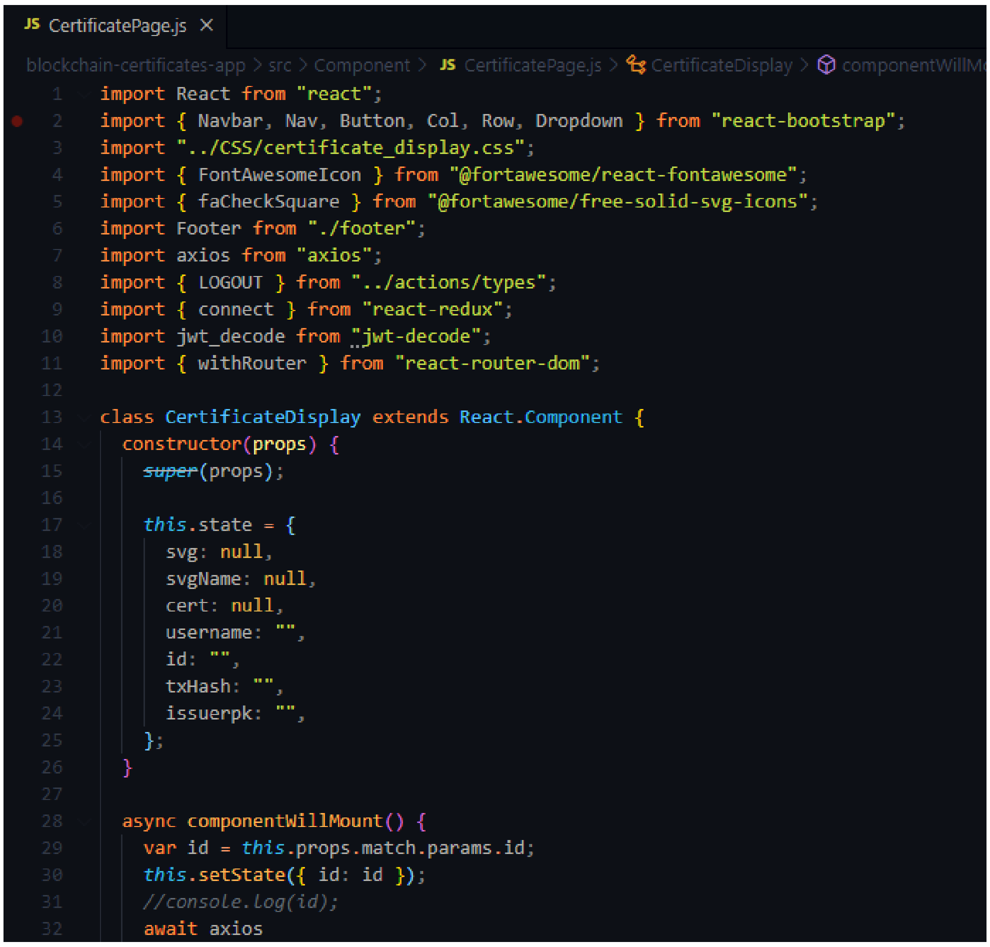
```
      await this.setState({ isAllowedToView: true });
    } else if (this.props.isAdmin) {
      this.setState({ isAllowedToView: true });
    } else {
      this.setState({ isAllowedToView: false });
      sessionStorage.removeItem("jwtToken");
      delete axios.defaults.headers.common["Authorization"];
      this.logout();
      this.props.history.push("/login");
    }
    console.log(Data.SVGN);
    await axios
      .get(
        `${process.env.REACT_APP_BACKEND_URL}api/static/media/${this.props.svgName}`,
        {
          responseType: "blob",
        }
      )
      .then((res) => {
        console.log(res.data);
        this.setState({ svg: URL.createObjectURL(res.data) });
      });
    // this.setState({
    //   svg: `${process.env.REACT_APP_BACKEND_URL}api/static/media/${this.props.match.params.svg}`,
    // });
  }

  updateSVG() {
```

Figure 5.5: Program for admin to add CSV file (b). The componentWillMount checks for admin authentication using the jwt token. If any unauthorized attempt is made, the user is redirected to the Login Page.


Admin_csv.js
index.js
App.js
StudentDashboard.js
CertificatePag
blockchain-certificates-app > src > Pages > Admin_csv.js > Admin_SVGUpload > updateSVG

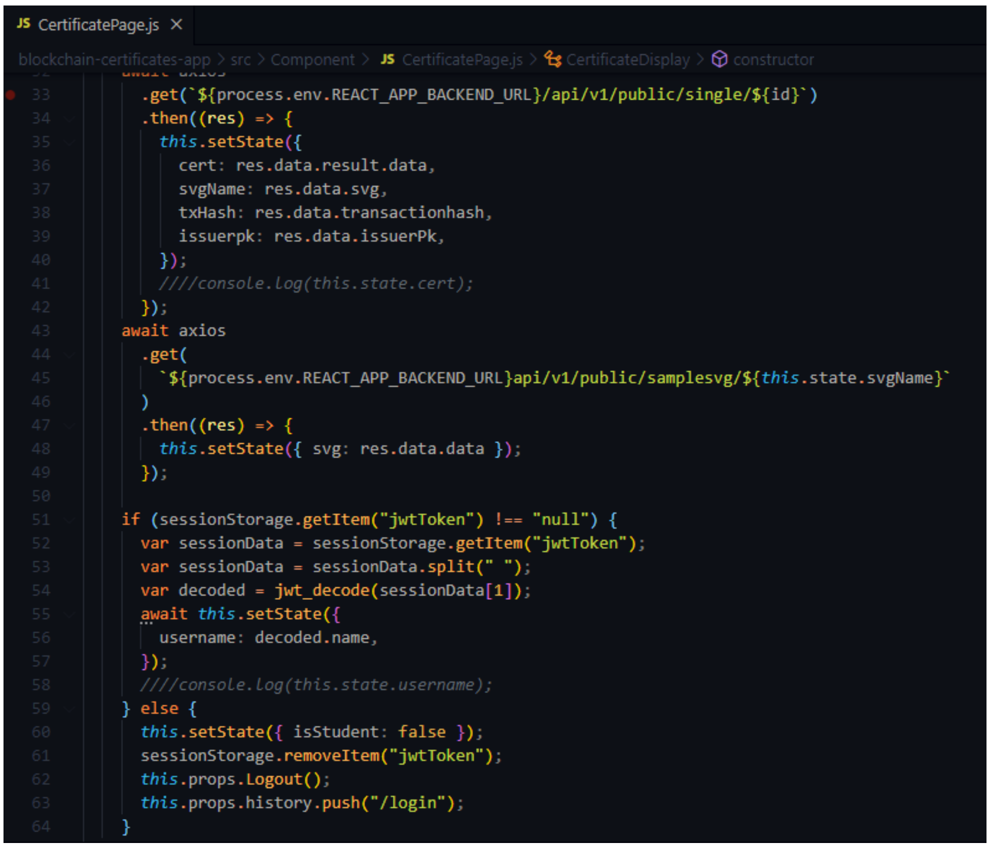
```
  updateSVG() {
    console.log(this.state.csv);
    // console.log(this.state.csv[0].data["student-name"]);
    var keys = Object.keys(this.state.csv[0].data);
    console.log(keys);
    var displaySVG = document.getElementById("SVG");
    var SVG = displaySVG.contentDocument;
    console.log(SVG);
    // console.log(SVG.getElementById("student-name").textContent);
    var keys = Object.keys(this.state.csv[0].data);
    console.log("Keys:", keys);


    for (var i = 0; i < keys.length; i++) {
      if (SVG.getElementById(keys[i]) !== null) {
        console.log("Done: ", keys[i]);
        SVG.getElementById(keys[i]).textContent = this.state.csv[i].data[
          keys[i]
        ];
      }
    }
  }
```

Figure 5.6: Program for admin to add CSV file (c). The updatetSVG function stores all the necessary data in the variables. The loop checks for null data in any row of the CSV and and stores the data in the state variables.



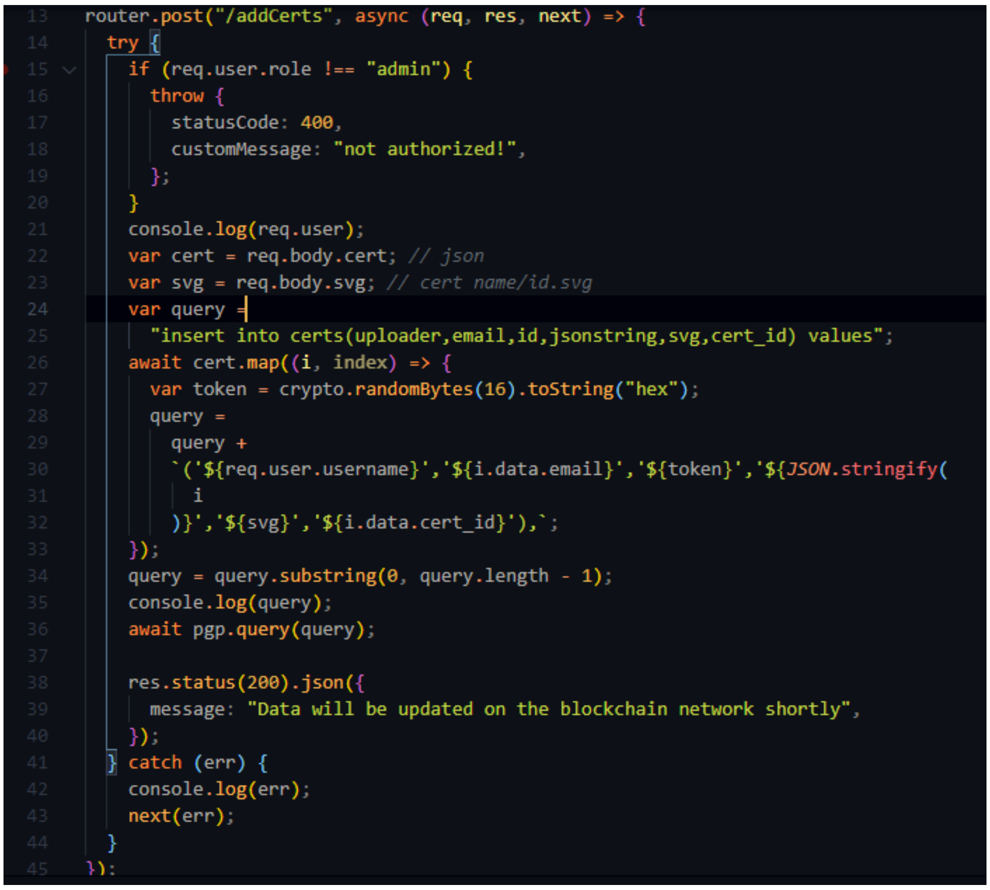
```
CertificatePage.js ×
blockchain-certificates-app > src > Component > CertificatePage.js > CertificateDisplay > componentWillM
import React from "react";
import { Navbar, Nav, Button, Col, Row, Dropdown } from "react-bootstrap";
import "../CSS/certificate_display.css";
import { FontAwesomeIcon } from "@fortawesome/react-fontawesome";
import { faCheckSquare } from "@fortawesome/free-solid-svg-icons";
import Footer from "./footer";
import axios from "axios";
import { LOGOUT } from "../actions/types";
import { connect } from "react-redux";
import jwt_decode from "jwt-decode";
import { withRouter } from "react-router-dom";

class CertificateDisplay extends React.Component {
  constructor(props) {
    super(props);

    this.state = {
      svg: null,
      svgName: null,
      cert: null,
      username: "",
      id: "",
      txHash: "",
      issuerpk: "",
    };
  }

  async componentWillMount() {
    var id = this.props.match.params.id;
    this.setState({ id: id });
    //console.log(id);
    await axios
```

Figure 5.7: Program for students to view their certificates (a). Importing the different UI components to render the page. The constructor variable initializes the state variables.


CertificatePage.js
blockchain-certificates-app > src > Component > CertificatePage.js > CertificateDisplay > constructor

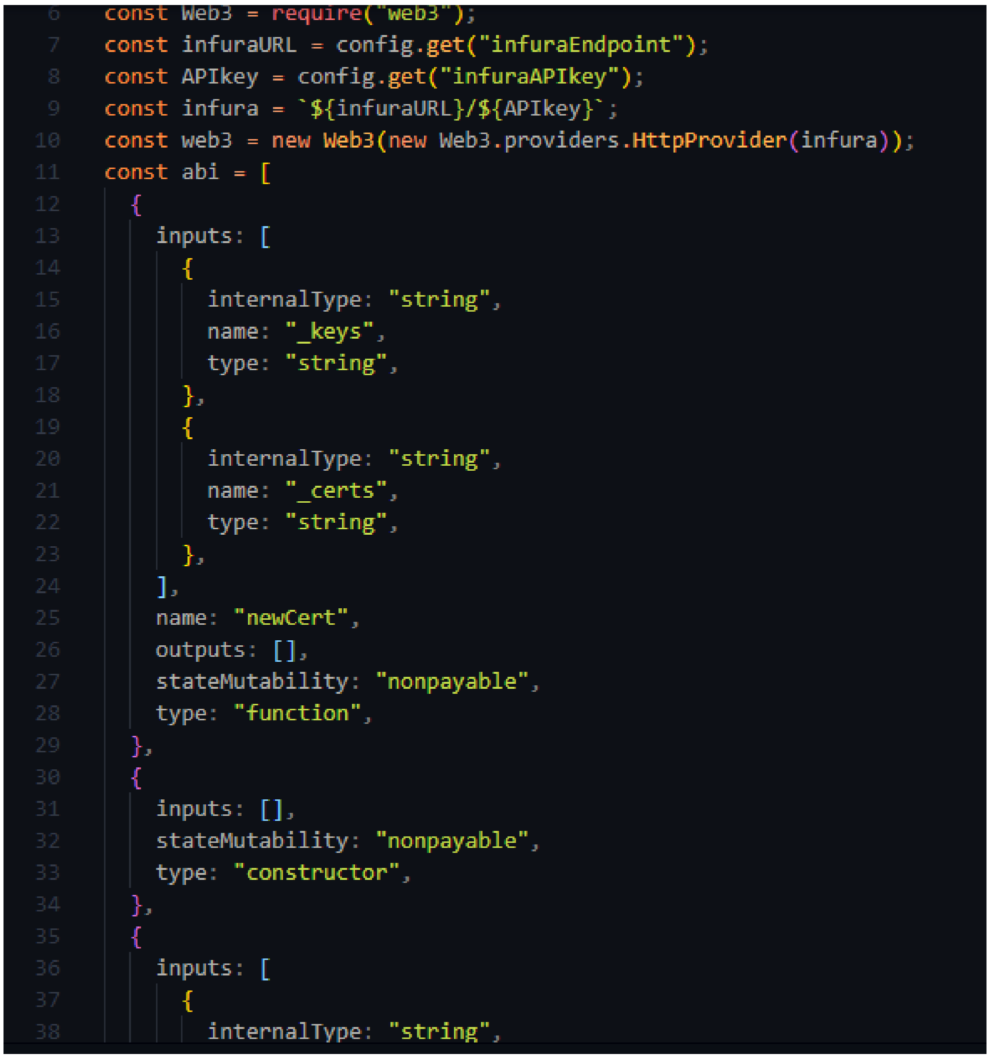
```
await axios
  .get(`${process.env.REACT_APP_BACKEND_URL}/api/v1/public/single/${id}`)
  .then((res) => {
    this.setState({
      cert: res.data.result.data,
      svgName: res.data.svg,
      txHash: res.data.transactionhash,
      issuerpk: res.data.issuerPk,
    });
    ////console.log(this.state.cert);
  });
await axios
  .get(
    `${process.env.REACT_APP_BACKEND_URL}api/v1/public/samplesvg/${this.state.svgName}`
  )
  .then((res) => {
    this.setState({ svg: res.data.data });
  });

if (sessionStorage.getItem("jwtToken") !== "null") {
  var sessionData = sessionStorage.getItem("jwtToken");
  var sessionData = sessionData.split(" ");
  var decoded = jwt_decode(sessionData[1]);
  await this.setState({
    username: decoded.name,
  });
  ////console.log(this.state.username);
} else {
  this.setState({ isStudent: false });
  sessionStorage.removeItem("jwtToken");
  this.props.Logout();
  this.props.history.push("/login");
}
```

Figure 5.8: Program for students to view their certificates (b). The componentWillMount function makes an API call to get the details of a single certificate fetching it by the Certificate ID. The svg file is then retrieved from the file system using the slug.



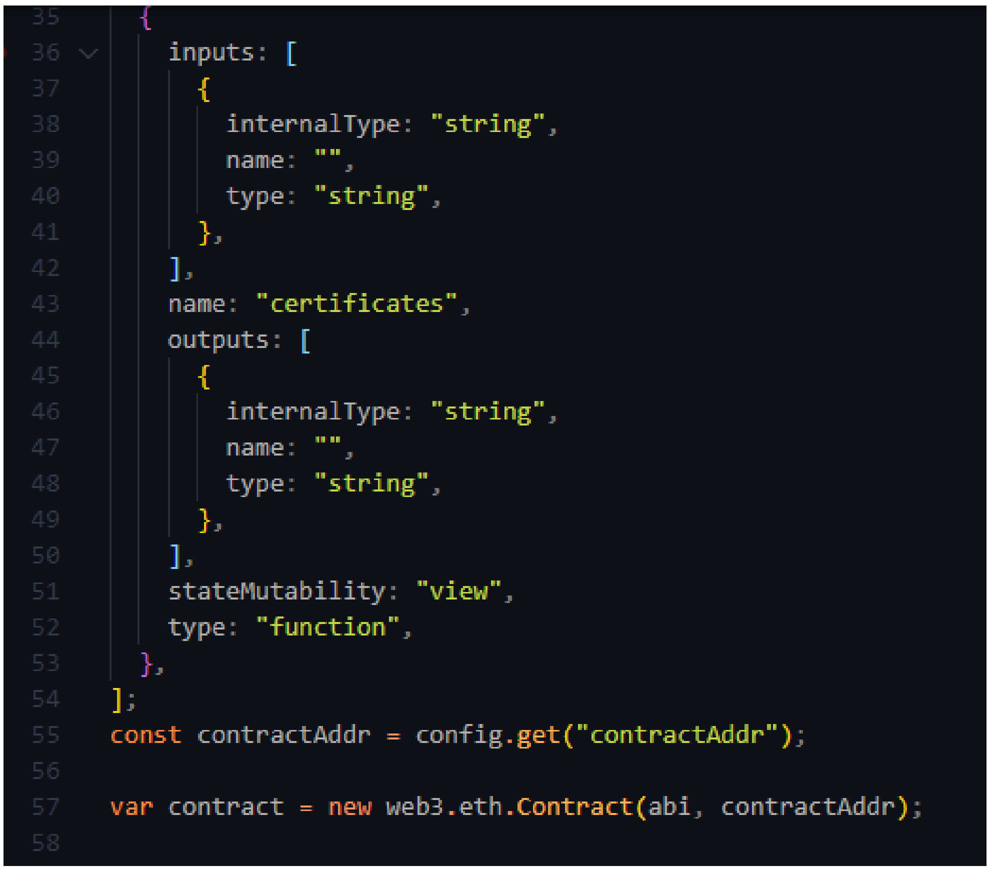
```
router.post("/addCerts", async (req, res, next) => {
  try {
    if (req.user.role !== "admin") {
      throw {
        statusCode: 400,
        customMessage: "not authorized!",
      };
    }
    console.log(req.user);
    var cert = req.body.cert; // json
    var svg = req.body.svg; // cert name/id.svg
    var query =
      "insert into certs(uploader,email,id,jsonstring,svg,cert_id) values";
    await cert.map((i, index) => {
      var token = crypto.randomBytes(16).toString("hex");
      query =
        query +
        `('${req.user.username}','${i.data.email}','${token}','${JSON.stringify(
          i
        )}','${svg}','${i.data.cert_id}'),`;
    });
    query = query.substring(0, query.length - 1);
    console.log(query);
    await pgp.query(query);

    res.status(200).json({
      message: "Data will be updated on the blockchain network shortly",
    });
  } catch (err) {
    console.log(err);
    next(err);
  }
});
```

Figure 5.9: Program to add certificate template in database. This route accepts the certificate data in json format and the template file in svg format. The data is stored in the database and awaits to be uploaded to BlockChain.



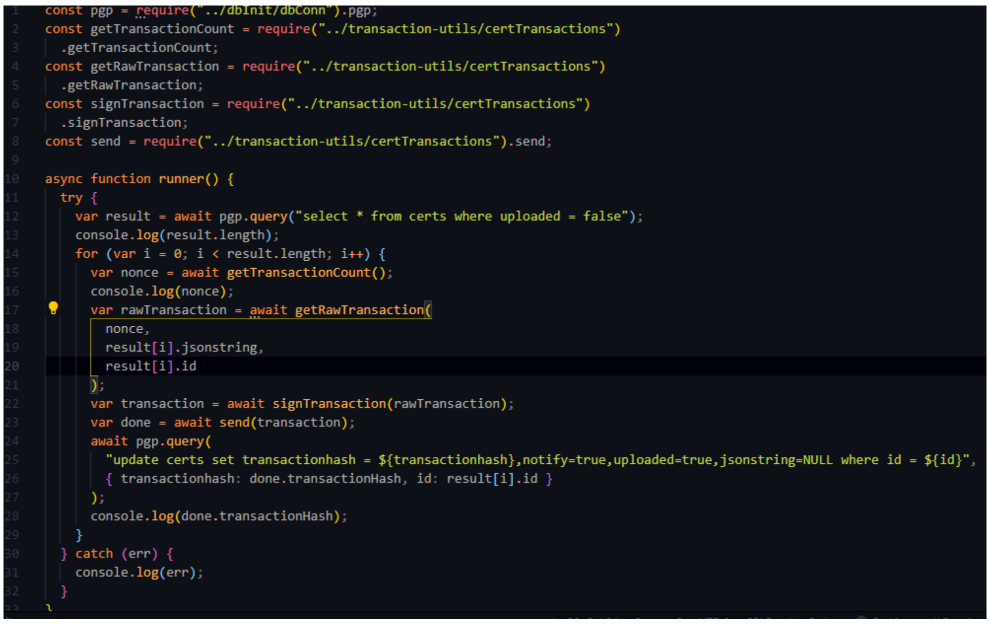
```
const Web3 = require("web3");
const infuraURL = config.get("infuraEndpoint");
const APIkey = config.get("infuraAPIkey");
const infura = `${infuraURL}/${APIkey}`;
const web3 = new Web3(new Web3.providers.HttpProvider(infura));
const abi = [
  {
    inputs: [
      {
        internalType: "string",
        name: "_keys",
        type: "string",
      },
      {
        internalType: "string",
        name: "_certs",
        type: "string",
      },
    ],
    name: "newCert",
    outputs: [],
    stateMutability: "nonpayable",
    type: "function",
  },
  {
    inputs: [],
    stateMutability: "nonpayable",
    type: "constructor",
  },
  {
    inputs: [
      {
        internalType: "string",
```

Figure 5.10: Connecting to the BlockChain network (a). The configurations to connect to the BlockChain network are set. The Infura helps connect the Etherium using an API. The web3 library is required. The abi contains the compiled Smart Contract.



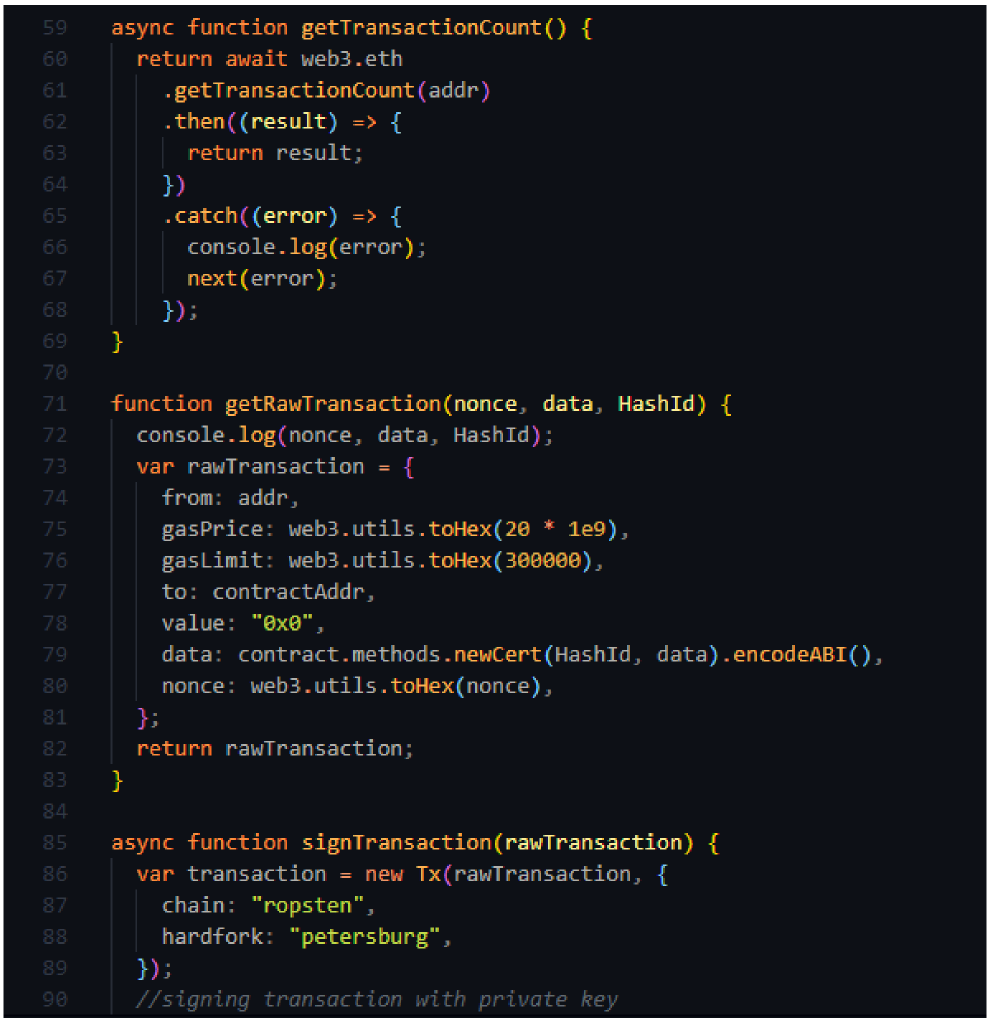
```
  {
    inputs: [
      {
        internalType: "string",
        name: "",
        type: "string",
      },
    ],
    name: "certificates",
    outputs: [
      {
        internalType: "string",
        name: "",
        type: "string",
      },
    ],
    stateMutability: "view",
    type: "function",
  },
];
const contractAddr = config.get("contractAddr");

var contract = new web3.eth.Contract(abi, contractAddr);

```

Figure 5.11: Connecting to the BlockChain network (b). The location where the Smart Contract is deployed is set in the contractAddr variable. The contract is accessed using the web3.eth.Contract method where the abi and address are set as parameters.



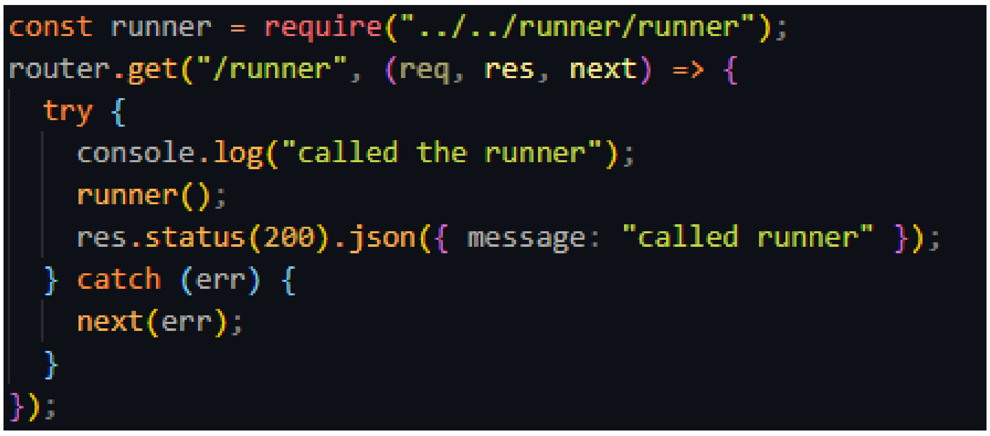
```javascript
const pgp = require("../dbInit/dbConn").pgp;
const getTransactionCount = require("../transaction-utils/certTransactions")
  .getTransactionCount;
const getRawTransaction = require("../transaction-utils/certTransactions")
  .getRawTransaction;
const signTransaction = require("../transaction-utils/certTransactions")
  .signTransaction;
const send = require("../transaction-utils/certTransactions").send;

async function runner() {
  try {
    var result = await pgp.query("select * from certs where uploaded = false");
    console.log(result.length);
    for (var i = 0; i < result.length; i++) {
      var nonce = await getTransactionCount();
      console.log(nonce);
      var rawTransaction = await getRawTransaction(
        nonce,
        result[i].jsonstring,
        result[i].id
      );
      var transaction = await signTransaction(rawTransaction);
      var done = await send(transaction);
      await pgp.query(
        "update certs set transactionhash = ${transactionhash},notify=true,uploaded=true,jsonstring=NULL where id = ${id}",
        { transactionhash: done.transactionHash, id: result[i].id }
      );
      console.log(done.transactionHash);
    }
  } catch (err) {
    console.log(err);
  }
}
```

Figure 5.12: Program to upload data on the blockchain network. The runner function is executed at mid-night to make transactions to BlockChain. The function looks for trnsactions that are not uploaded to BlockChain from the database and fetches them. Once the data is uploaded to BlockChain using the web3 library, the database is updated to set the transaction hash.



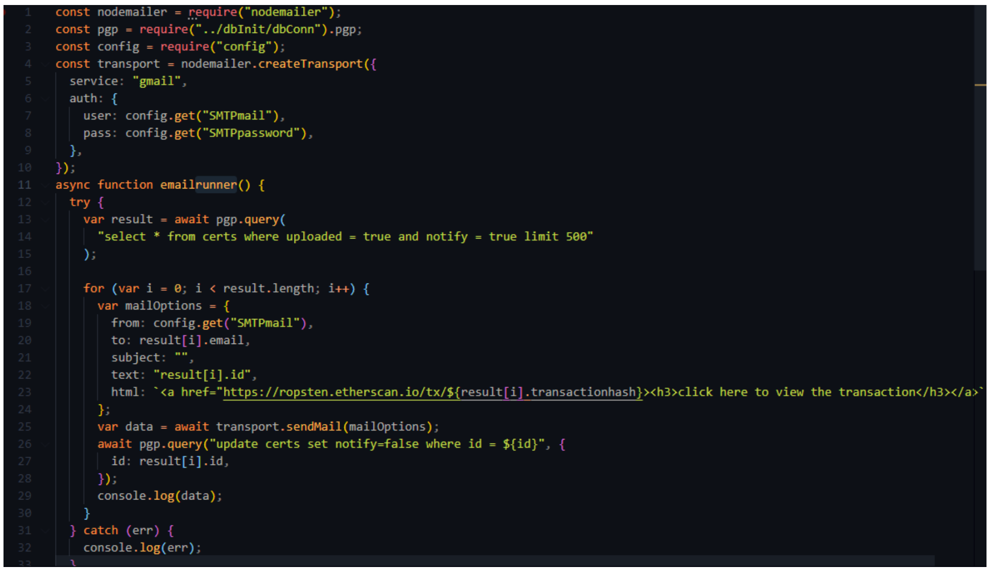
```
async function getTransactionCount() {
  return await web3.eth
    .getTransactionCount(addr)
    .then((result) => {
      return result;
    })
    .catch((error) => {
      console.log(error);
      next(error);
    });
}

function getRawTransaction(nonce, data, HashId) {
  console.log(nonce, data, HashId);
  var rawTransaction = {
    from: addr,
    gasPrice: web3.utils.toHex(20 * 1e9),
    gasLimit: web3.utils.toHex(300000),
    to: contractAddr,
    value: "0x0",
    data: contract.methods.newCert(HashId, data).encodeABI(),
    nonce: web3.utils.toHex(nonce),
  };
  return rawTransaction;
}

async function signTransaction(rawTransaction) {
  var transaction = new Tx(rawTransaction, {
    chain: "ropsten",
    hardfork: "petersburg",
  });
  //signing transaction with private key
```

Figure 5.13: Methods used to verify, get information and transaction hash number from the blockchain network. The getTransactionCount method gets the number of transactions done on the contract which is further used as the nonce value of the transaction, The getRaTransaction method creates the raw transaction with the details of the sender and the data.



```javascript
const runner = require("../../runner/runner");
router.get("/runner", (req, res, next) => {
  try {
    console.log("called the runner");
    runner();
    res.status(200).json({ message: "called runner" });
  } catch (err) {
    next(err);
  }
});
```

Figure 5.14: Program used to upload data periodically on the blockchain network ensuring no data is being dropped or skipped.



```javascript
const nodemailer = require("nodemailer");
const pgp = require("../dbInit/dbConn").pgp;
const config = require("config");
const transport = nodemailer.createTransport({
  service: "gmail",
  auth: {
    user: config.get("SMTPmail"),
    pass: config.get("SMTPpassword"),
  },
});
async function emailrunner() {
  try {
    var result = await pgp.query(
      "select * from certs where uploaded = true and notify = true limit 500"
    );

    for (var i = 0; i < result.length; i++) {
      var mailOptions = {
        from: config.get("SMTPmail"),
        to: result[i].email,
        subject: "",
        text: "result[i].id",
        html: `<a href="https://ropsten.etherscan.io/tx/${result[i].transactionhash}><h3>click here to view the transaction</h3></a>`
      };
      var data = await transport.sendMail(mailOptions);
      await pgp.query("update certs set notify=false where id = ${id}", {
        id: result[i].id,
      });
      console.log(data);
    }
  } catch (err) {
    console.log(err);
```

Figure 5.15: Program used to send the transaction value or the unique key of the certificate to the appropriate owner of the certificate once it has been uploaded on the blockchain network. The email is sent to the owner of the certificate which contains the link to the certificate. The link can be used to directly fetch the certificate and can be shared with anyone.

Chapter 6

Testing

6.0.1 Methodology used

White box testing is implemented in this project to verify the flow of I/O and to improve the usability and security. This testing methodology involves the all round testing of the software code. Using White Box testing all the internal security holes have been patched and the flow of specific inputs through the code are checked. The testing for this project is majorly done at the unit level with all the individual modules tested one by one to verify a working flow for an application.

6.0.2 Advantages

Code optimization by finding hidden errors. White box tests cases can be easily automated. Testing is more thorough as all code paths are usually covered. Testing can start early in SDLC even if GUI is not available.

6.0.3 Module-wise testing

Test No	Test Name	Expected Result	Actual Result
1	Register new user	Register user to the database	Registered successfully
2	Login as admin	Open admin dashboard	Logged in successfully
3	Login as student	Open student dashboard	Logged in successfully
4	Upload SVG template of certificate	Save SVG template with proper serial number into the database	Saved successfully
5	Upload CSV data onto the blockchain network	Save CSV data on blockchain network with template serial number	Saved successfully
6	Receive mail	Mail to be received when certificate is validated	Mail received successfully
7	Retrieve data from blockchain	Retrieve appropriate certificate data requested by student	Retrieved data successfully
8	Mapping the retrieved data	Map the data retrieved from the blockchain on the certificate template	Mapping of data successfully

Table 6.1: Testing different modules of the system

Chapter 7

Result

The following images are the results/output screenshots of our finished application "Educhain".

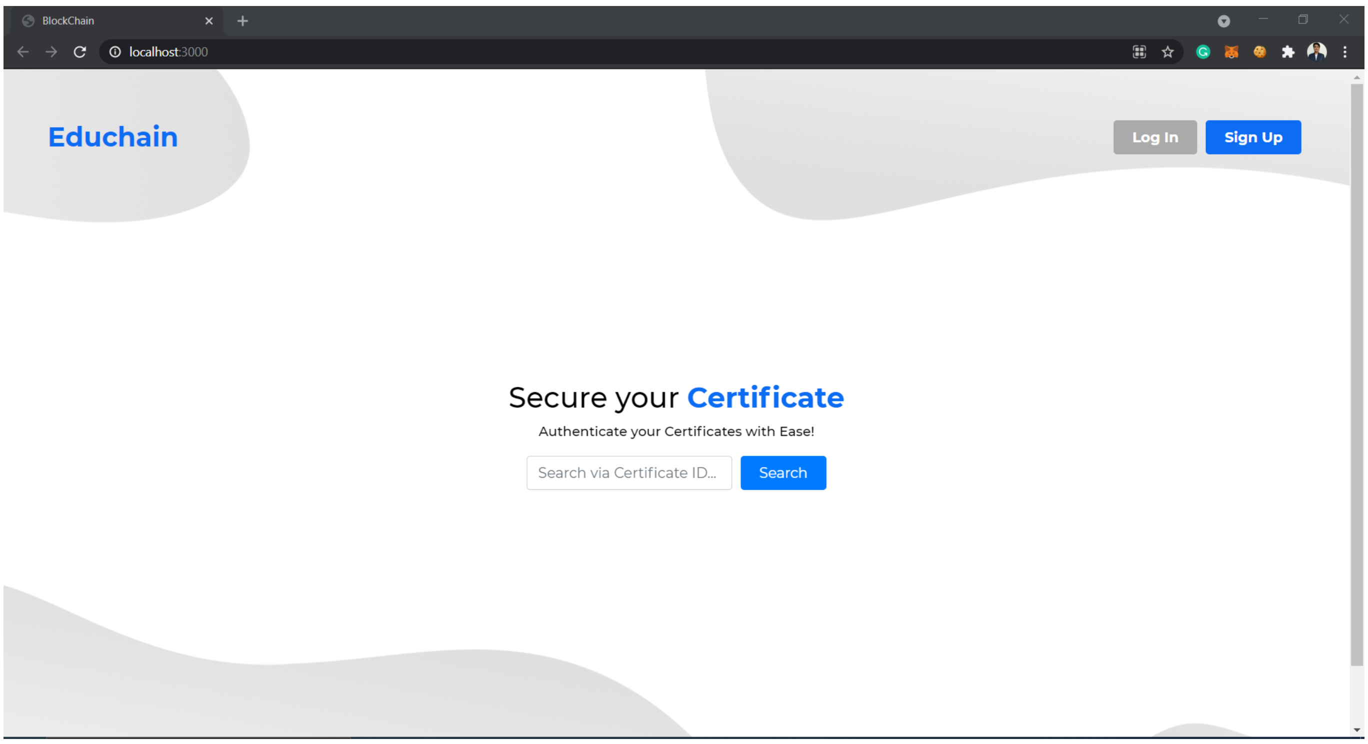


Figure 7.1: Home screen.

This is the landing screen of our project. As seen, at the top right it has an option to Login/Sign-Up. Coming to the center of the screen we have a search field, this field is to search a authenticated certificate, also it can be used by Institutions/Universities to verify the certificates.

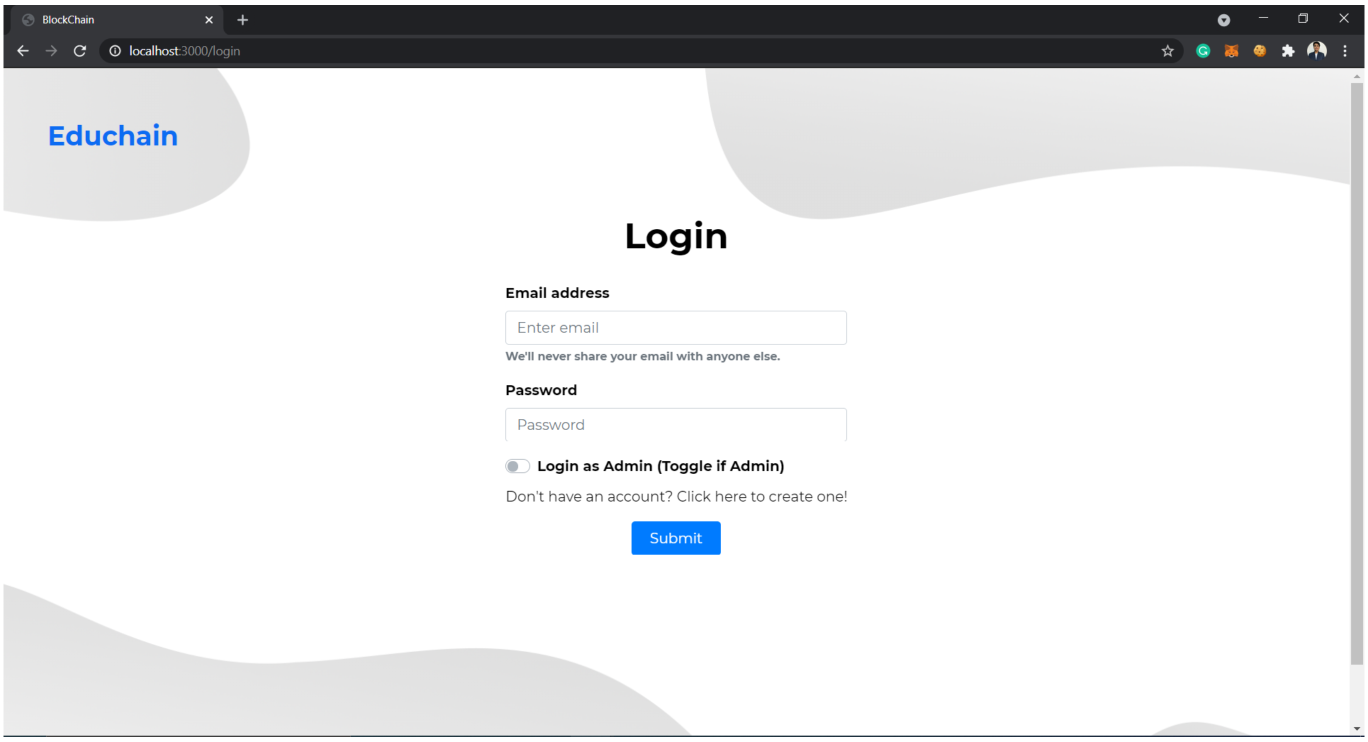
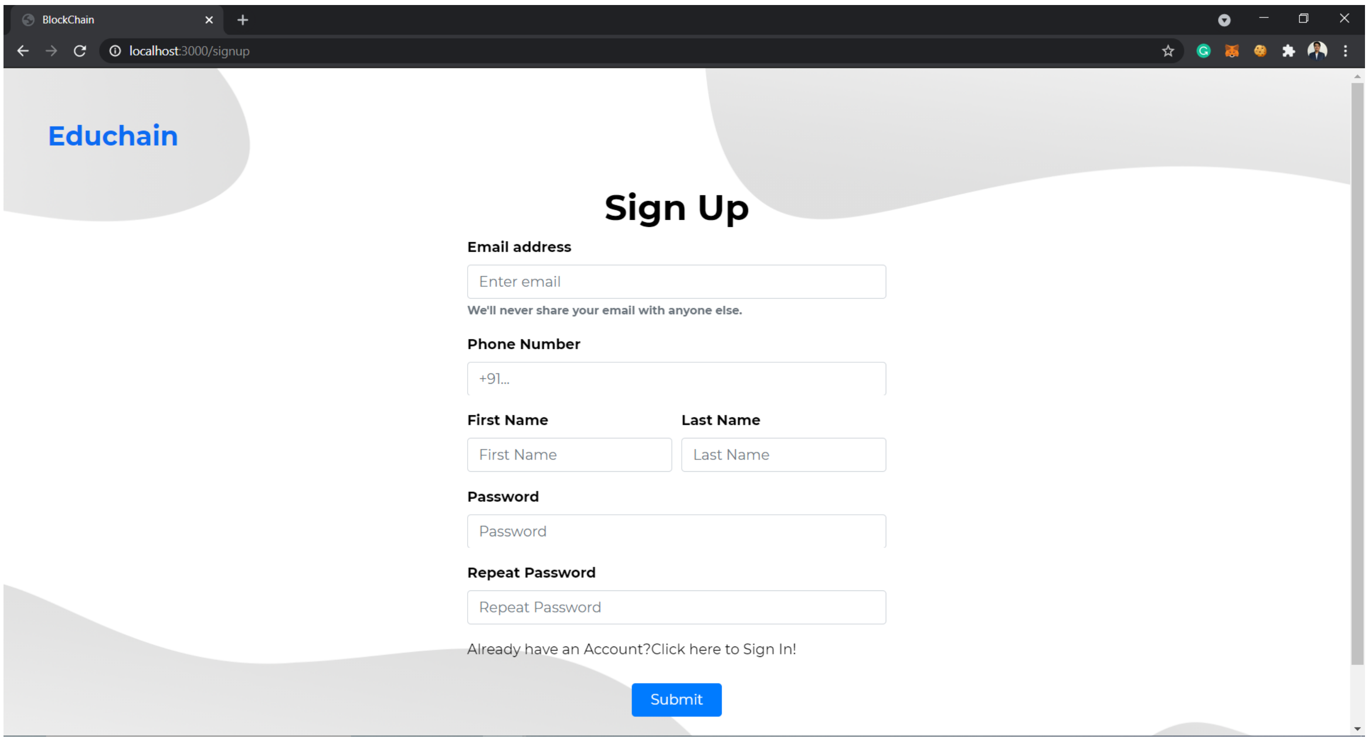

BlockChain
localhost:3000/login
Educhain
Login
Email address
Enter email
We'll never share your email with anyone else.
Password
Password
Login as Admin (Toggle if Admin)
Don't have an account? Click here to create one!
Submit

Figure 7.2: Login page.

Following the landing screen, the login page comes next. Here an existing user can log himself/herself into the system to access the utilities of the system. If an Adimn wants to login then that admin has to toggle on the "Login as Admin" after filling the credentials.



Sign Up

Email address

Enter email

We'll never share your email with anyone else.

Phone Number

+91...

First Name	Last Name
First Name	Last Name

Password

Password

Repeat Password

Repeat Password

Already have an Account?Click here to Sign In!

Submit

Figure 7.3: Registration page.
This is the Sign-Up page where new users can register themselves into the system. A new user need to register with their Email address, Phone number, Name & Password. After filling the required fields, the user will be registered to the system.

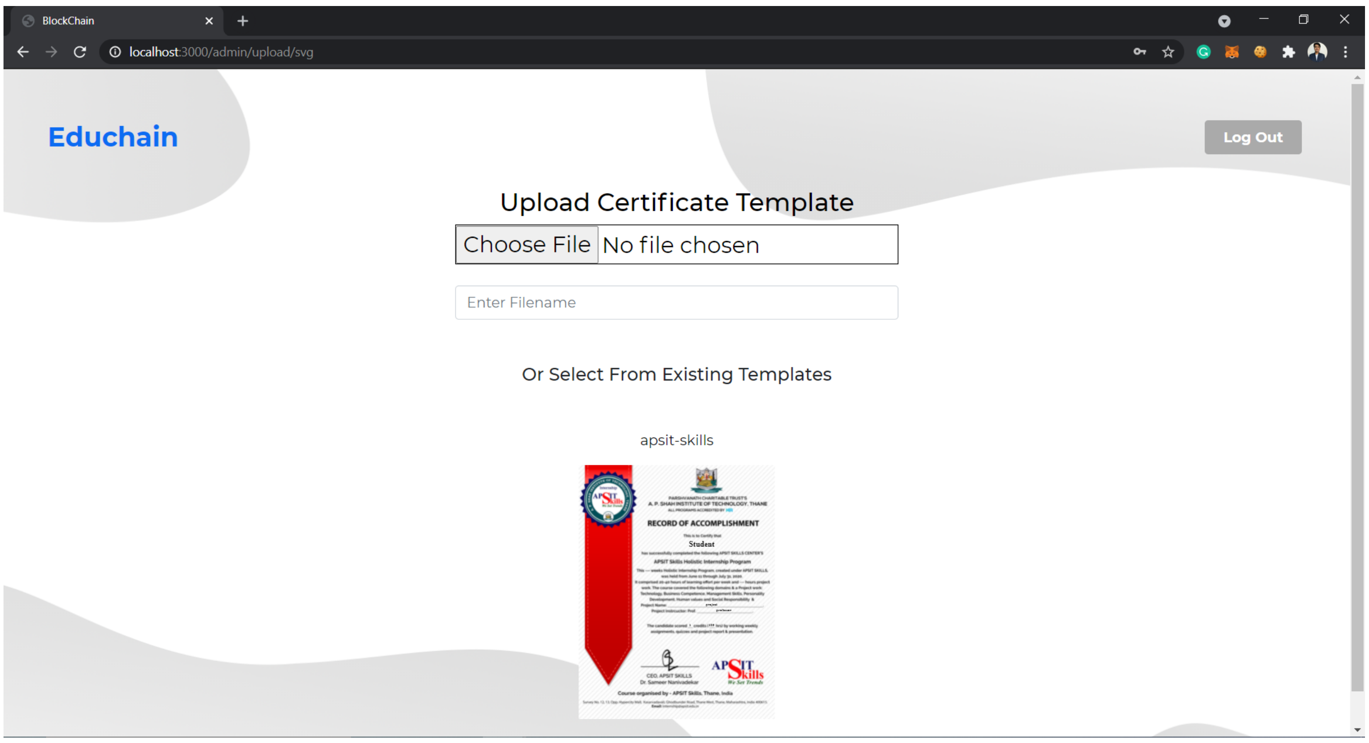
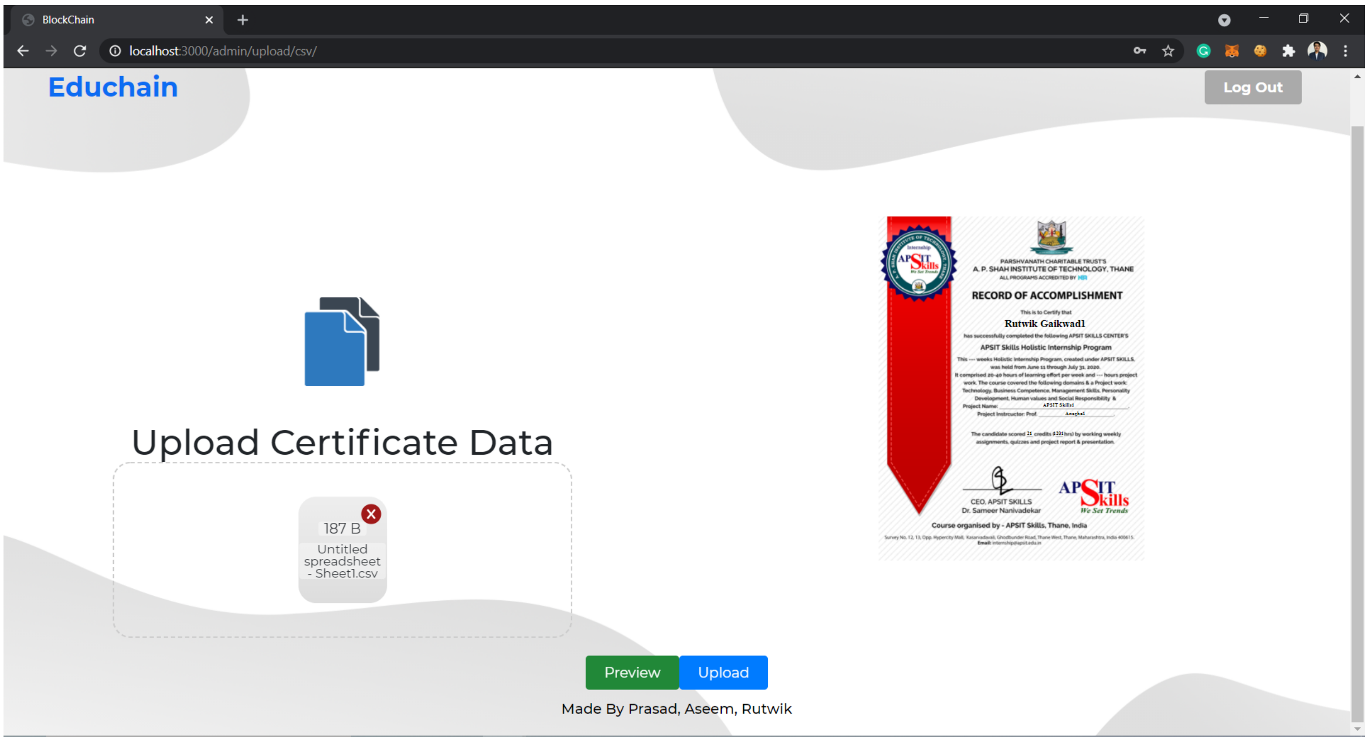


Figure 7.4: Admin portal to add certificate template in SVG format.

After logging in as an admin, the admin can choose to create a certificate. To do so, first the admin has to upload a certificate template in SVG format. As we can see in the Figure 7.4, the admin can choose the certificate template from his/her file system by clicking on "Choose File" and also give a name to the template. The Admin can also choose to use the existing templates that were used in the past.


BlockChain
localhost:3000/admin/upload/csv/
Educhain
Log Out
Upload Certificate Data
187 B
Untitled spreadsheet - Sheet1.csv
Preview
Upload
Made By Prasad, Aseem, Rutwik

PARSHVANATH CHARITABLE TRUST'S
A. P. SHAH INSTITUTE OF TECHNOLOGY, THANE
ALL PROGRAMS ACCREDITED BY NBA

RECORD OF ACCOMPLISHMENT

This is to Certify that
Rutwik Gaikwad1
has successfully completed the following APSIT SKILLS CENTER'S
APSIT Skills Holistic Internship Program
This --- weeks Holistic Internship Program, created under APSIT SKILLS, was held from June 11 through July 31, 2020.
It comprised 20-40 hours of learning effort per week and --- hours project work. The course covered the following domains & a Project work: Technology, Business Competence, Management Skills, Personality Development, Human values and Social Responsibility &
Project Name: APSIT Skills1.
Project Instrcuctor: Prof. Anagha1.

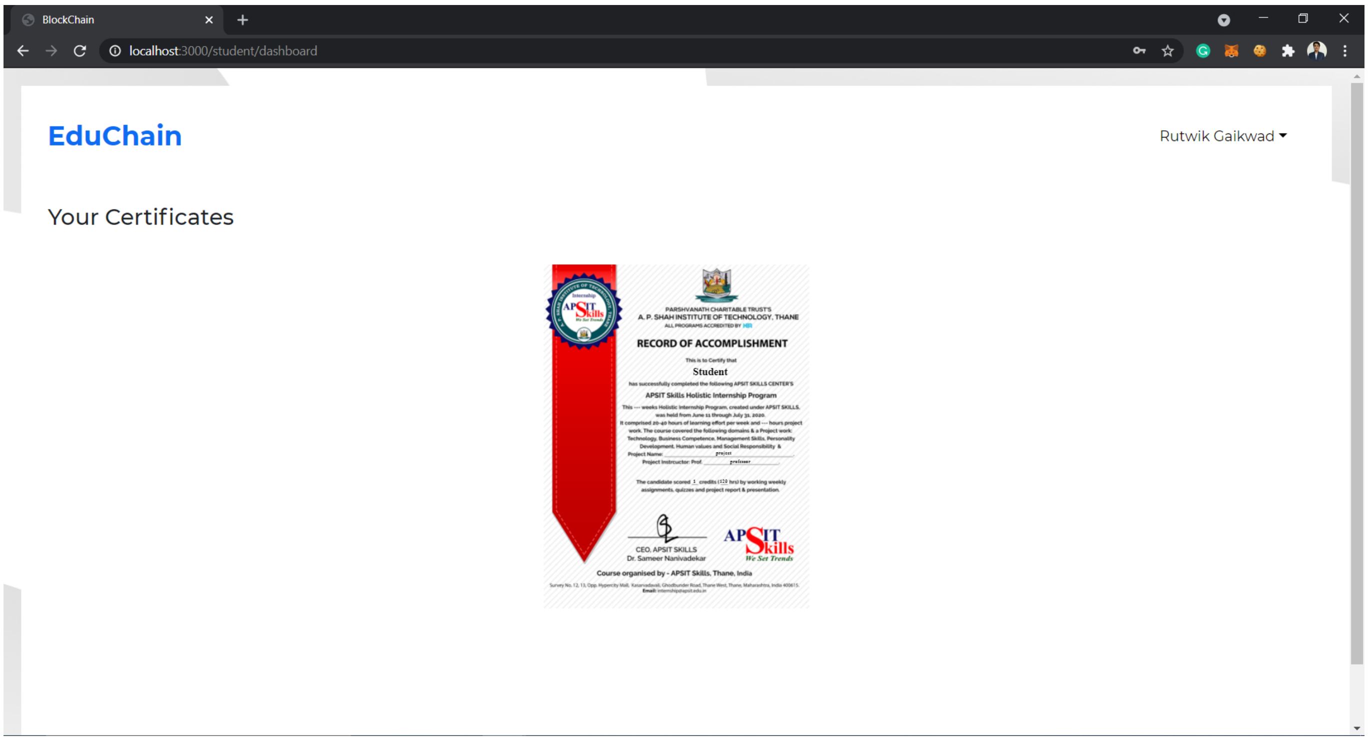
The candidate scored 21 credits (1201 hrs) by working weekly assignments, quizzes and project report & presentation.

CEO, APSIT SKILLS
Dr. Sameer Nanivadekar

APSIT Skills We Set Trends

Course organised by - APSIT Skills, Thane, India
Survey No. 12, 13, Opp. Hypercity Mall, Kasarvadavali, Ghodbunder Road, Thane West, Thane, Maharashtra, India 400615.
Email: internship@apsit.edu.in

Figure 7.5: Admin portal to add CSV file for the selected template with the preview option. Figure 7.5 Shows the page where the Admin can add CSV file(which contains the student details) for the selected certificate template. This Page also gives the preview of the template with student details on it.



BlockChain

localhost:3000/student/dashboard

EduChain

Rutwik Gaikwad

Your Certificates

PARSHVANATH CHARITABLE TRUST'S
A. P. SHAH INSTITUTE OF TECHNOLOGY, THANE
ALL PROGRAMS ACCREDITED BY NBA

RECORD OF ACCOMPLISHMENT

This is to Certify that

Student

has successfully completed the following APSIT SKILLS CENTER'S

APSIT Skills Holistic Internship Program

This --- weeks Holistic Internship Program, created under APSIT SKILLS, was held from June 11 through July 31, 2020. It comprised 20-40 hours of learning effort per week and --- hours project work. The course covered the following domains & a Project work: Technology, Business Competence, Management Skills, Personality Development, Human values and Social Responsibility &

Project Name: project

Project Instrcuctor: Prof. professor

The candidate scored 1 credits (120 hrs) by working weekly assignments, quizzes and project report & presentation.

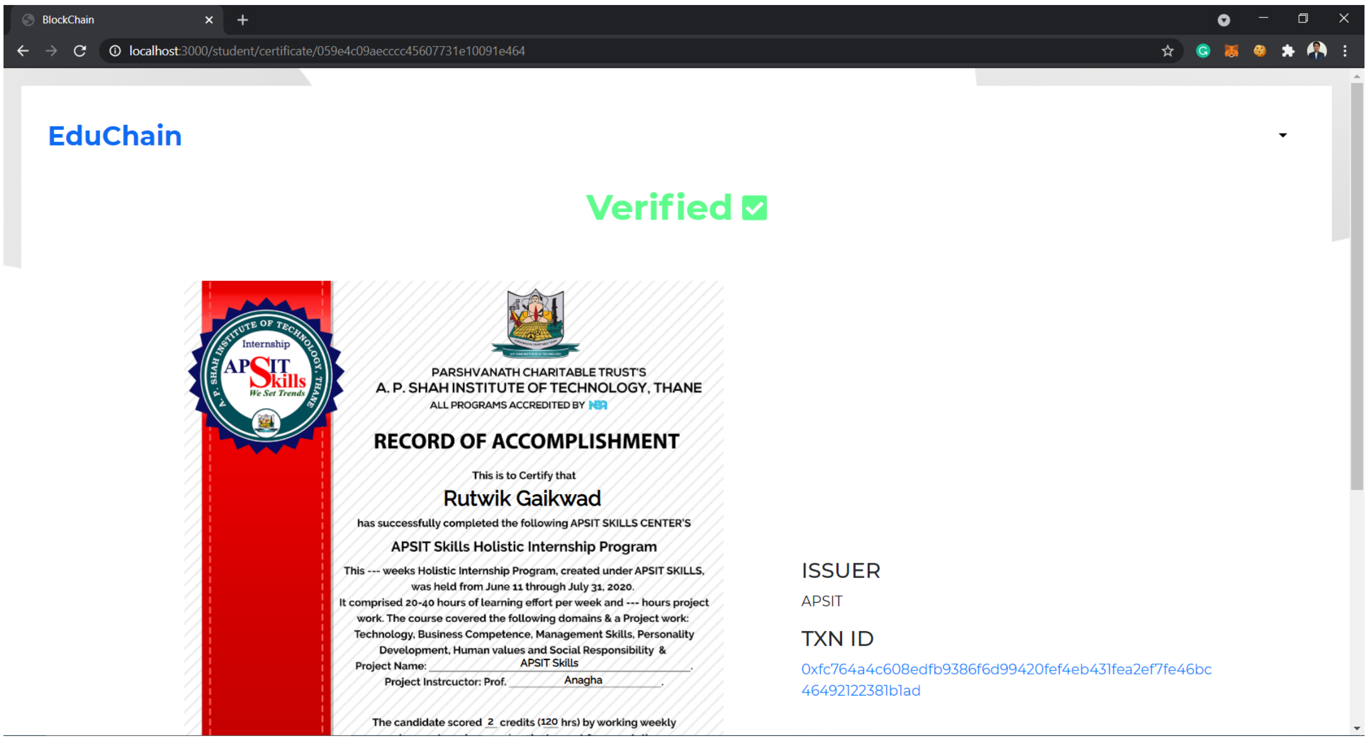
CEO, APSIT SKILLS
Dr. Sameer Nanivadekar

Course organised by - APSIT Skills, Thane, India

Survey No. 12, 13, Opp. Hypercity Mall, Kasarvadavali, Ghodbunder Road, Thane West, Thane, Maharashtra, India 400615.
Email: internship@apsit.edu.in

Figure 7.6: Dashboard for Students to view all Certificates

After logging in as Student, the student has an option to view certificated authenticated to them. Figure 7.6 shows the dashboard that shows all the certificates authenticated to a student.


BlockChain

localhost:3000/student/certificate/059e4c09aecccc45607731e10091e464

EduChain

Verified ✓

PARSHVANATH CHARITABLE TRUST'S
A. P. SHAH INSTITUTE OF TECHNOLOGY, THANE
ALL PROGRAMS ACCREDITED BY NBA

RECORD OF ACCOMPLISHMENT

This is to Certify that
Rutwik Gaikwad
has successfully completed the following APSIT SKILLS CENTER'S
APSIT Skills Holistic Internship Program
This --- weeks Holistic Internship Program, created under APSIT SKILLS, was held from June 11 through July 31, 2020.
It comprised 20-40 hours of learning effort per week and --- hours project work. The course covered the following domains & a Project work: Technology, Business Competence, Management Skills, Personality Development, Human values and Social Responsibility &
Project Name: APSIT Skills.
Project Instrcuctor: Prof. Anagha.

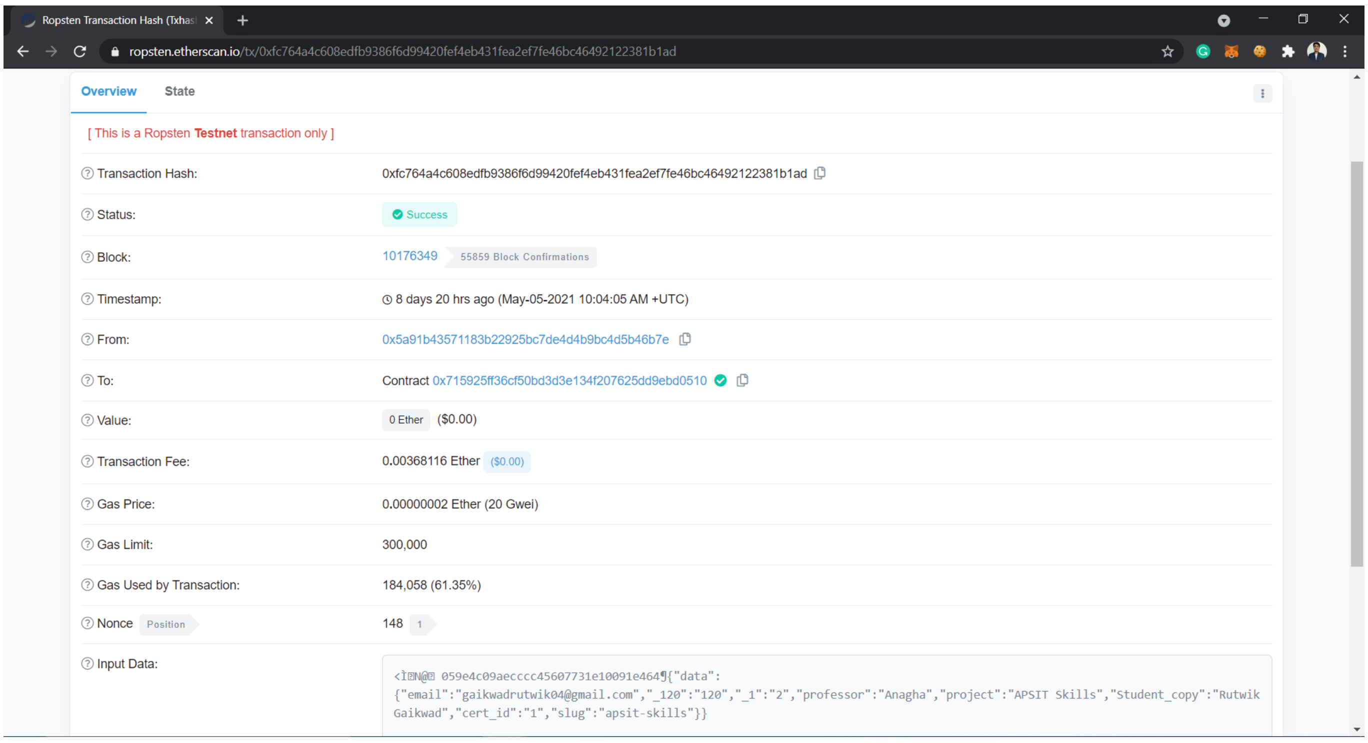
The candidate scored 2 credits (120 hrs) by working weekly

ISSUER
APSIT

TXN ID
0xfc764a4c608edfb9386f6d99420fef4eb431fea2ef7fe46bc46492122381b1ad

Figure 7.7: Display of Individual Certificate with verification from BlockChain

When we search a certificate on the landing screen with the certificate's unique id we can see the details of the certificate which contain the name of the issuer and also the blockchain transaction Hash.


Ropsten Transaction Hash (Txhas

ropsten.etherscan.io/tx/0xfc764a4c608edfb9386f6d99420fef4eb431fea2ef7fe46bc46492122381b1ad

Overview State

[This is a Ropsten **Testnet** transaction only]

Transaction Hash:	0xfc764a4c608edfb9386f6d99420fef4eb431fea2ef7fe46bc46492122381b1ad
Status:	Success
Block:	10176349 55859 Block Confirmations
Timestamp:	8 days 20 hrs ago (May-05-2021 10:04:05 AM +UTC)
From:	0x5a91b43571183b22925bc7de4d4b9bc4d5b46b7e
To:	Contract 0x715925ff36cf50bd3d3e134f207625dd9ebd0510
Value:	0 Ether ($0.00)
Transaction Fee:	0.00368116 Ether ($0.00)
Gas Price:	0.00000002 Ether (20 Gwei)
Gas Limit:	300,000
Gas Used by Transaction:	184,058 (61.35%)
Nonce Position	148 1
Input Data:	

```
<Ì N@ 059e4c09aecccc45607731e10091e464¶{"data":
{"email":"gaikwadrutwik04@gmail.com","_120":"120","_1":"2","professor":"Anagha","project":"APSIT Skills","Student_copy":"Rutwik
Gaikwad","cert_id":"1","slug":"apsit-skills"}}
```

Figure 7.8: Verify transaction done on BlockChain

After clicking on the block transaction hash from Figure 7.7, the website is redirected to the ”robsten.etherscan.io” page which shows the details of the complete transaction on the blockchain of that specific certificate.

Chapter 8

Conclusions and Future Scope

8.0.1 Conclusion

In this paper, we have successfully proposed a system where blockchain technology can be used to store and retrieve certificate data. The project will help companies issue certificates securely through Blockchain and can be verified by anyone with the unique link/code to each certificate. The system uses the concept of SVG templates for the certificates which would be stored on a local server and to store data over the blockchain for secure and reliable storage. This will minimize the cost of storing the entire certificate on the blockchain network. Storing only the data of the certificate will minimize cost and thereby turn out to be cost-efficient. The only drawback of this system is that the template of the certificate needs to be properly created with the SVG's text area ID to match with the header of the CSV file. Only a properly crafted SVG and CSV pair will result in proper certificate generation through Blockchain.

8.0.2 Future Scope

1. Currently, the certificate template which is stored on local file storage is the weakest link in the system. The template relies on the security of the File System used. The use of IPFS - InterPlanetary File System can secure the certificate template stored thereby adding to the security of the system.

2. IPFS has the capacity to store files over Blockchain allowing secure storage and retrieval of the certificate template.

3. The system can be further extended to store other online documents of importance to ensure the integrity of data and documents being stored securely.

Bibliography

[1] UK: NARIC Survey: universities and education fraud – 75% cannot spot a fake certificate - Feb 2020, https://uknaric.org/2020/02/26/survey-universities-and-education-fraud-75-cannot-spot-a-fake-certificate/

[2] BBC File on 4 exposes a multi-million pound global trade in fake diplomas - Jan 2018, https://www.bbc.co.uk/programmes/b09ly731

[3] Blockchain, https://en.wikipedia.org/wiki/Blockchain.

[4] Smart Contracts, https://www.ibm.com/blogs/blockchain/2018/07/what-are-smart-contracts-on-blockchain/

[5] Scalable Vector Graphics, https://en.wikipedia.org/wiki/Scalable_Vector_Graphics

[6] Shanmuga Priya R, Swetha N. Online Certificate Validation Using Blockchain. Published in Int. Jnl. Of Advanced Networking Applications (IJANA)

[7] Nitin Kumavat, Swapnil Mengade, Dishant Desai, JesalVarolia (2019). Certificate Verification System using Blockchain. International Journal for Research in Applied Science Engineering Technology (IJRASET).

[8] Oliver Miquel, Moreno Joan,Prieto Gerson, Ben tez David (2018). Using blockchain as a tool for tracking and verification of official degrees: business model 29th European Regional Conference of the International Telecommunications Society (ITS): "Towards a Digital Future: Turning Technology into Markets?", Trento, Italy, 1st - 4th August 2018.

[9] K. Kuvshinov, I. Nikiforov, J. Mostovoy, and D. Mukhutdinov, "Disciplina: Blockchain for Education," pp. 1–17, 2018.

[10] D.T.T. Anh, M. Zhang, B.C. Ooi, and G. Chen, "Untangling Blockchain: A Data Processing View of Blockchain Systems," IEEE Trans. Knowl. Data Eng., vol. 4347, no. c, pp. 1–20, 2018.

[11] Kumar, NM Saravana. quot;Implementation of artificial intelligence in imparting education and evaluating student performance.&quot; Journal of Artificial Intelligence 1, no. 01 (2019): 1-9.

[12] Smys, S., Joy Iong Zong Chen, and Subarna Shakya. quot;Survey on Neural Network Architectures with Deep Learning.&quot; Journal of Soft Computing Paradigm (JSCP) 2, no. 03 (2020): 186-194

Acknowledgement

We have great pleasure in presenting the report on **Online Certificate Generation Verification using Blockchain Framework** We take this opportunity to express our sincere thanks towards our guide **Prof. Kiran Deshpande** Department of IT, APSIT thane for providing the technical guidelines and suggestions regarding line of work. We would like to express our gratitude towards his constant encouragement, support and guidance through the development of project.

We thank **Prof. Kiran B. Deshpande** Head of Department,IT, APSIT for his encouragement during progress meeting and providing guidelines to write this report.

We thank **Prof. Vishal S. Badgujar** BE project co-ordinator, Department of IT, APSIT for being encouraging throughout the course and for guidance.

We also thank the entire staff of APSIT for their invaluable help rendered during the course of this work. We wish to express our deep gratitude towards all our colleagues of APSIT for their encouragement.

Prasad Jadhav:
17104003:

Rutwik Gaikwad:
17104074:

Aseem Godambe:
17104058:

Publication

Paper entitled **"Online Certificate Generation & Verification using Blockchain Framework"** is presented at **"ICSCS 2021 : International Conference on Soft Computing for Security Applications"** by **Prasad Jadhav**, **Rutwik Gaikwad**, **Aseem Godambe** and **Kiran Deshpande**.