# A PROJECT REPORT ON

**MASS MAIL-DISPACTHER**

# Project Description:

The Mass Mail Dispatcher project is a sophisticated email automation system designed to revolutionize the way organizations send bulk emails to their target audience The mass mail dispatcher project addresses this challenge by providing a centralized platform that automates and streamlines the email distribution process. The primary objective of the mass mail dispatcher project is to develop a robust and user-friendly software tool that allows users to manage and send bulk emails effortlessly. The project team focuses on creating a feature-rich application that offers essential The mass mail dispatcher often incorporates advanced scheduling capabilities, allowing users to set specific dates and times.

# Usage Instructions:

**To use the Mass Mail Dispatcher project, follow these steps:**

**1. Upload a CSV file containing email addresses.**

**2. The system will validate and sort the email addresses, separating the valid and invalid ones.**

**3. Enter the email subject, body, and other details.**

**4. Click on the send button to dispatch the emails**

# Technology Stack:

**The Mass Mail Dispatcher project uses the following technologies:**

**• HTML**

**• CSS**

**• JavaScript**

**• EmailJS API**

# Examples:

**Here are some examples of how the Mass Mail Dispatcher project can be used:**

**➤ Sending newsletters to a list of subscribers.**

**➤ Sending event invitations to a list of attendees.**

**➤ Sending product updates to a list of customers**

# IMPLEMENTATION:

**1.Scheduling and Automation**: Implement scheduling capabilities that enable users to set specific dates and times for email campaigns. Develop features for recurring campaigns, drip campaigns, or triggered emails based on specific events or user actions.

**2.Email Delivery**: Integrate with an email server or SMTP provider to handle the actual delivery of emails. Implement mechanisms to handle email queuing, throttling, and error handling. Configure DNS records (SPF, DKIM, DMARC) to improve email deliverability and prevent emails from being flagged as spam.

**3.Testing and Quality Assurance**: Conduct comprehensive testing, including unit testing, integration testing, and system testing. Test different scenarios and edge cases to ensure the system functions as intended. Perform performance testing to evaluate the system's capacity to handle high email volumes

# CODE

!DOCTYPE html>

<html>

<head>

<title>Send Mail</title>

<script src="https://cdn.emailjs.com/dist/email.min.js"></script>

<link rel="stylesheet" href="style.css">

<link href="https://fonts.googleapis.com/css?family=Raleway:200,100,400" rel="stylesheet" type="text/css" />

</head>

<body>

<h1>

<span class="txt-rotate" data-period="2000"

data-rotate='[ "&nbsp&nbsp Hi !! friends", "Want to send ", "Multiple Emails", "&nbsp in One - click", "Then Welcome..","You are on","Right Platform..","Just upload ","&nbsp &nbsp CSV file","Write message","&nbsp &nbsp &nbsp And ","&nbsp Send Emails","&nbsp &nbsp &nbsp ENJOY !!!" ]'></span>

</h1>

<p id="heading">Mass Mail<br> Dispatcher</p>

<form method="post">

<label for="senderEmail">From :</label>

<input type="email" id="senderEmail" name="senderEmail" required>

<br><br><br>

<label for="subject" style="margin-top: -2em;z-index: 22;position: absolute;">Subject:</label>

<textarea id="subject" name="subject" required></textarea><br><br><br><br><label for="csvFile">CSV File:</label>

<input type="file" id="csvFile" name="csvFile" accept=".csv" required><br><br>

<label for="message">Message:</label>

<textarea id="message" name="message" required></textarea>

<br><br>

<input type="button" value="Send Emails" style="font-weight: bold;" onclick="sendEmails()">

<br>

<div style="display:flex">

<div>

<p style="color: #393E41;">Valid Emails: <span id="validEmailCount"></span></p>

<div id="validEmails" style="float: left"></div>

</div>

<div style="margin-left: 15px">

<p style="color: #393E41;">Invalid Emails: <span id="invalidEmailCount"></span></p>

<div id="invalidEmails" style="float: left"></div>

</div>

</div>

</form>

<span><img src="Images/boy.png" alt="" class="img1"></span>

<img src="Images/cloud.png" alt="" class="img2">

<div class="word"></div>

<script type="text/javascript">

(function () {

emailjs.init("5EpzE1X9S3UEAOSn1"); // replace with your actual user ID

})();

SENDING EMAILS

function sendEmails() {

var senderEmail = document.getElementById("senderEmail").value;

var message = document.getElementById("message").value;

var subject = document.getElementById("subject").value;

var validEmails = [];

var invalidEmails = [];

// Read contents of CSV file

var file = document.getElementById("csvFile").files[0];

var reader = new FileReader();

reader.readAsText(file);

reader.onload = function (event) {

var csv = event.target.result;

var lines = csv.split('\n');

for (var i = 0; i < lines.length; i++) {

var email = lines[i].trim();

var emailRegex = /^[\w-\.]+@([\w-]+\.)+[\w-]{2,3}$/

;

if (emailRegex.test(email)) {

validEmails.push(email);

} else {

invalidEmails.push(email);

}

}

// Send email to valid email addresses

for (var j = 0; j < validEmails.length; j++) {

var templateParams = {

to\_name: validEmails[j],

from\_name: senderEmail,

message\_html: message,

subject\_html: subject

};

// Replace you Service ID ↓ and Template ID ↓ here.

emailjs.send('service\_13s3n3a', 'template\_huwrmzf', templateParams)

.then(function (response) {

console.log("SUCCESS", response);

}, function (error) {

console.log("FAILED", error);

});

}

alert("Emails sent to valid email addresses.");

};

}

/Thinking/

var TxtRotate = function (el, toRotate, period) {

this.toRotate = toRotate;

this.el = el;

this.loopNum = 0;

this.period = parseInt(period, 1) || 1000;

this.txt = '';

this.tick();

this.isDeleting = false;

};

TxtRotate.prototype.tick = function () {

var i = this.loopNum % this.toRotate.length;

var fullTxt = this.toRotate[i];

if (this.isDeleting) {

this.txt = fullTxt.substring(0, this.txt.length - 1);

} else {

this.txt = fullTxt.substring(0, this.txt.length + 1);

}

this.el.innerHTML = '<span class="wrap">' + this.txt + '</span>';

var that = this;

var delta = 300 - Math.random() \* 100;

if (this.isDeleting) { delta /= 2; }

if (!this.isDeleting && this.txt === fullTxt) {

delta = this.period;

this.isDeleting = true;

} else if (this.isDeleting && this.txt === '') {

this.isDeleting = false;

this.loopNum++;

delta = 500;

}

setTimeout(function () {

that.tick();

}, delta);

};

window.onload = function () {

var elements = document.getElementsByClassName('txt-rotate');

for (var i = 0; i < elements.length; i++) {

var toRotate = elements[i].getAttribute('data-rotate');

var period = elements[i].getAttribute('data-period');

if (toRotate) {

new TxtRotate(elements[i], JSON.parse(toRotate), period);

}

}

// INJECT CSS

var css = document.createElement("style");

css.type = "text/css";

css.innerHTML = ".txt-rotate > .wrap { border-right: 0.08em solid #666 }";

document.body.appendChild(css);

};

</script>

<script type="text/javascript">

document.getElementById("csvFile").addEventListener("change", function () {

var validEmails = [];

var invalidEmails = [];

// Read contents of CSV file

var file = document.getElementById("csvFile").files[0];

var reader = new FileReader();

reader.readAsText(file);

reader.onload = function (event) {

var csv = event.target.result;

var lines = csv.split('\n');

for (var i = 0; i < lines.length; i++) {

var email = lines[i].trim();

var emailRegex = /^[\w-\.]+@([\w-]+\.)+[\w-]{2,3}$/ ;

if (emailRegex.test(email)) {

validEmails.push(email);

} else {

invalidEmails.push(email);

}

}

// Display valid and invalid emails

document.getElementById("validEmails").innerHTML = validEmails.join("<br><br>");

document.getElementById("invalidEmails").innerHTML = invalidEmails.join("<br><br>");

document.getElementById("validEmailCount").innerText = "(" + validEmails.length + ")";

document.getElementById("invalidEmailCount").innerText = "(" + invalidEmails.length + ")";

};

});

// Display valid and invalid emails

document.getElementById("validEmails").innerHTML = validEmails.join("<br><br>");

document.getElementById("invalidEmails").innerHTML = invalidEmails.join("<br><br>");

document.getElementById("validEmailCount").innerText = "(" + validEmails.length + ")";

document.getElementById("invalidEmailCount").innerText = "(" + invalidEmails.length + ")";

};

});

</script>

</body>

</html>

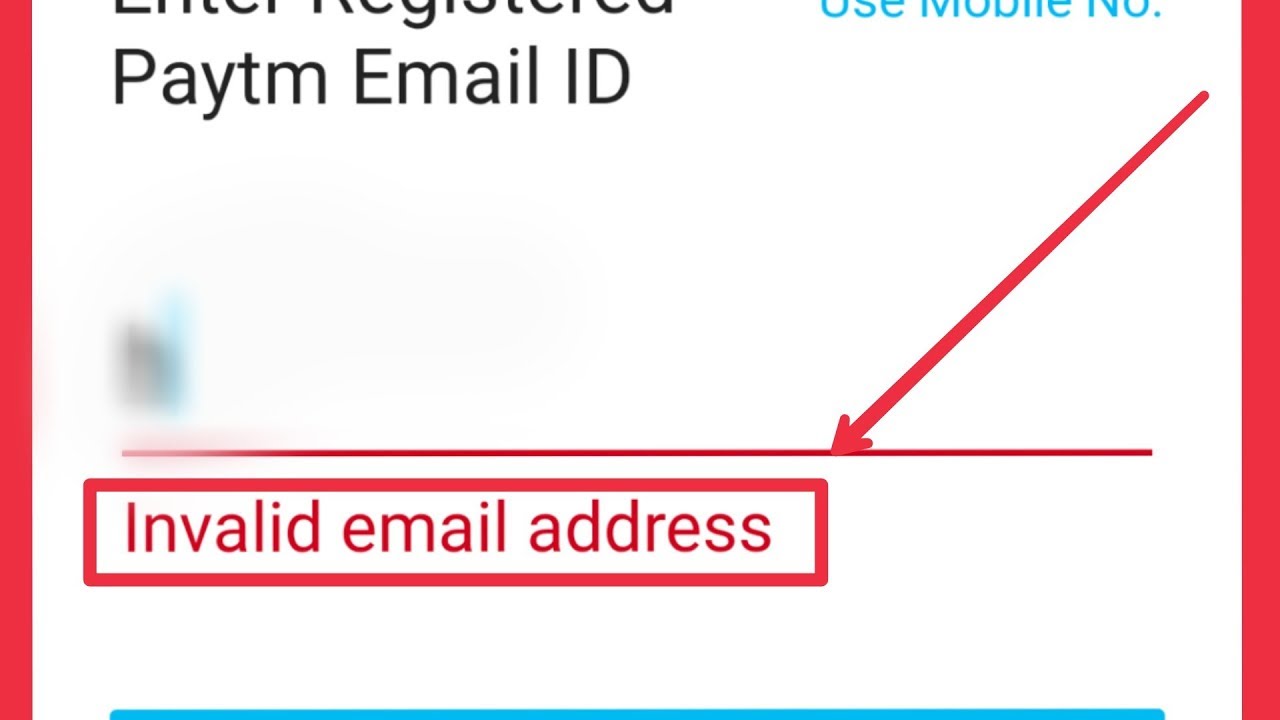
SAMPLE SCREENSHOT:



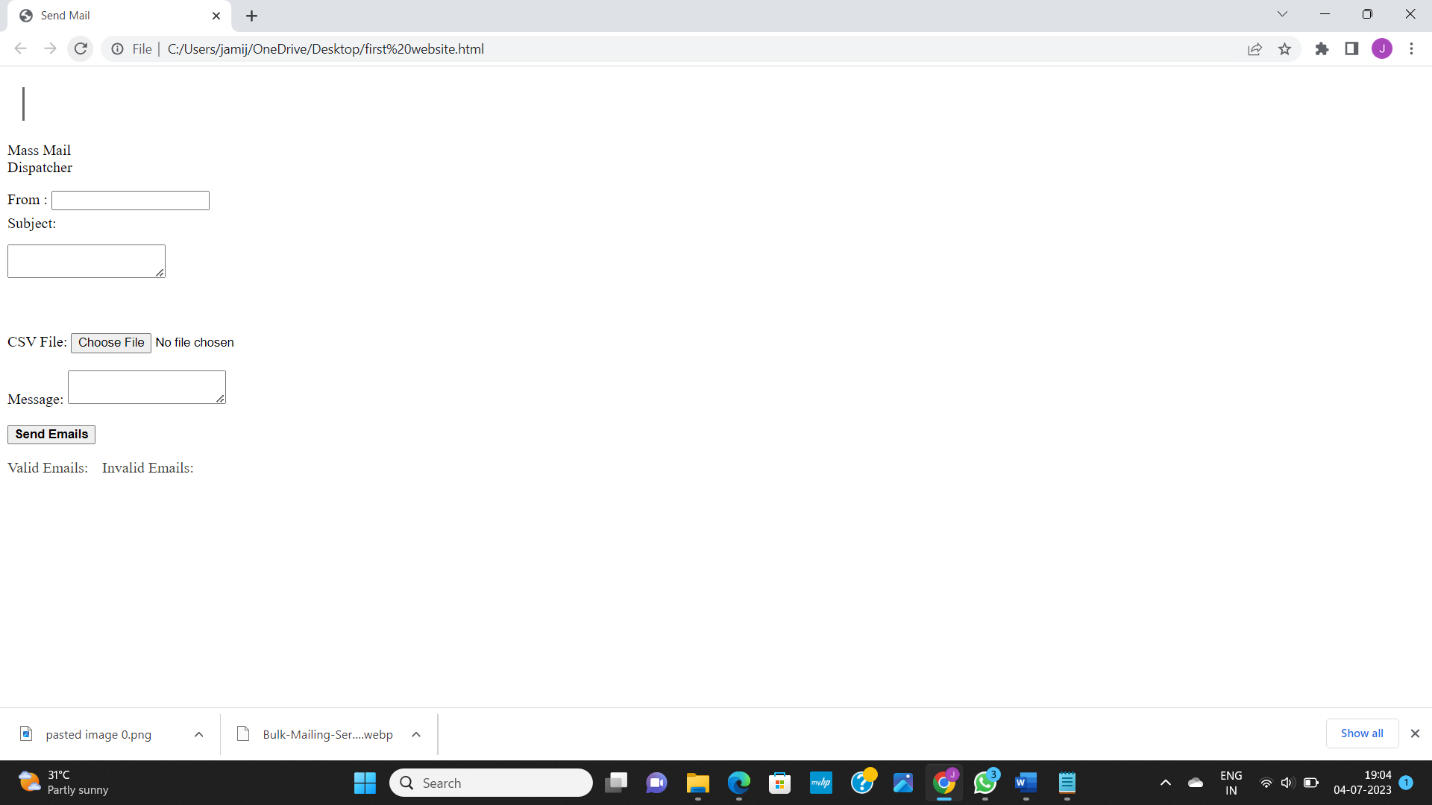
VAILD:



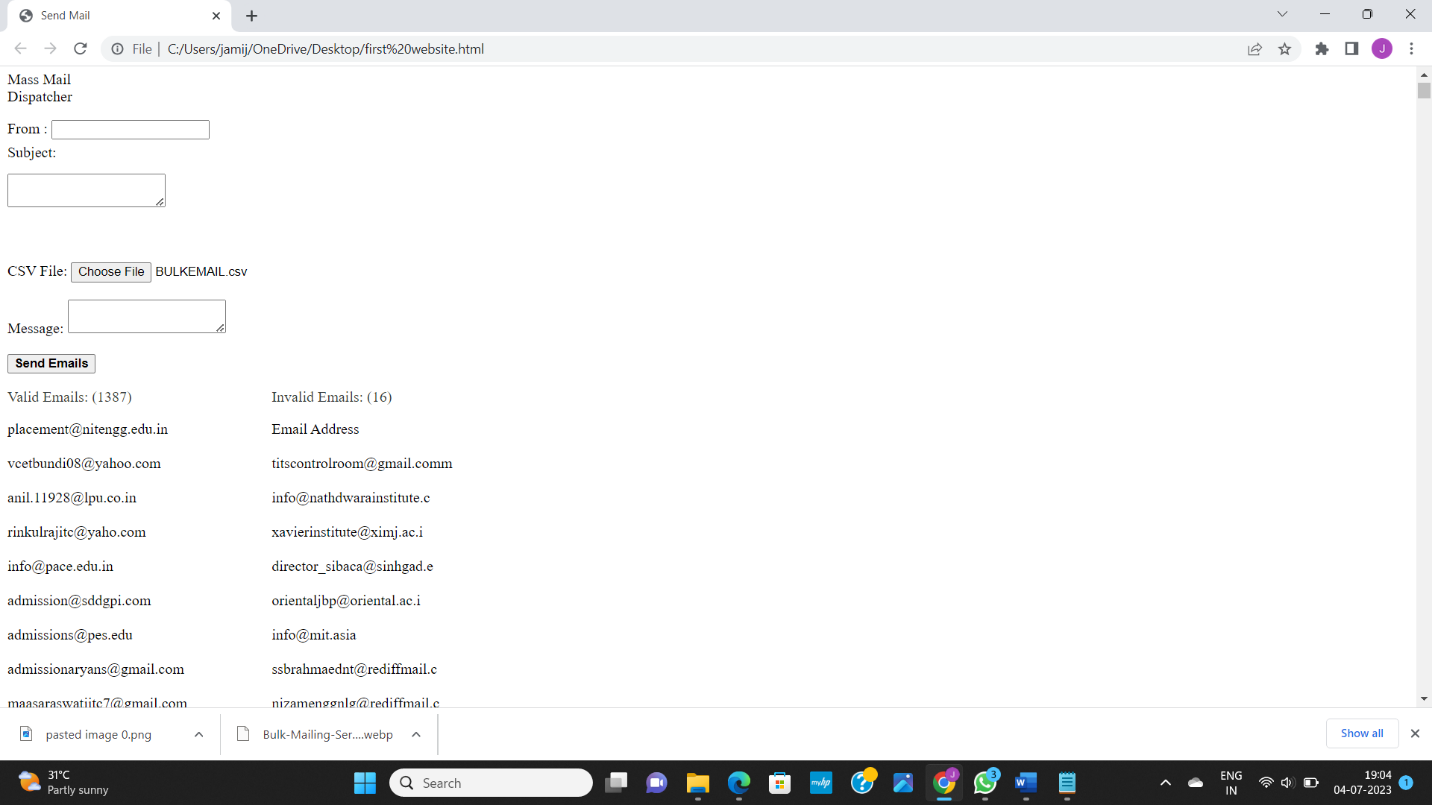
INVALID:



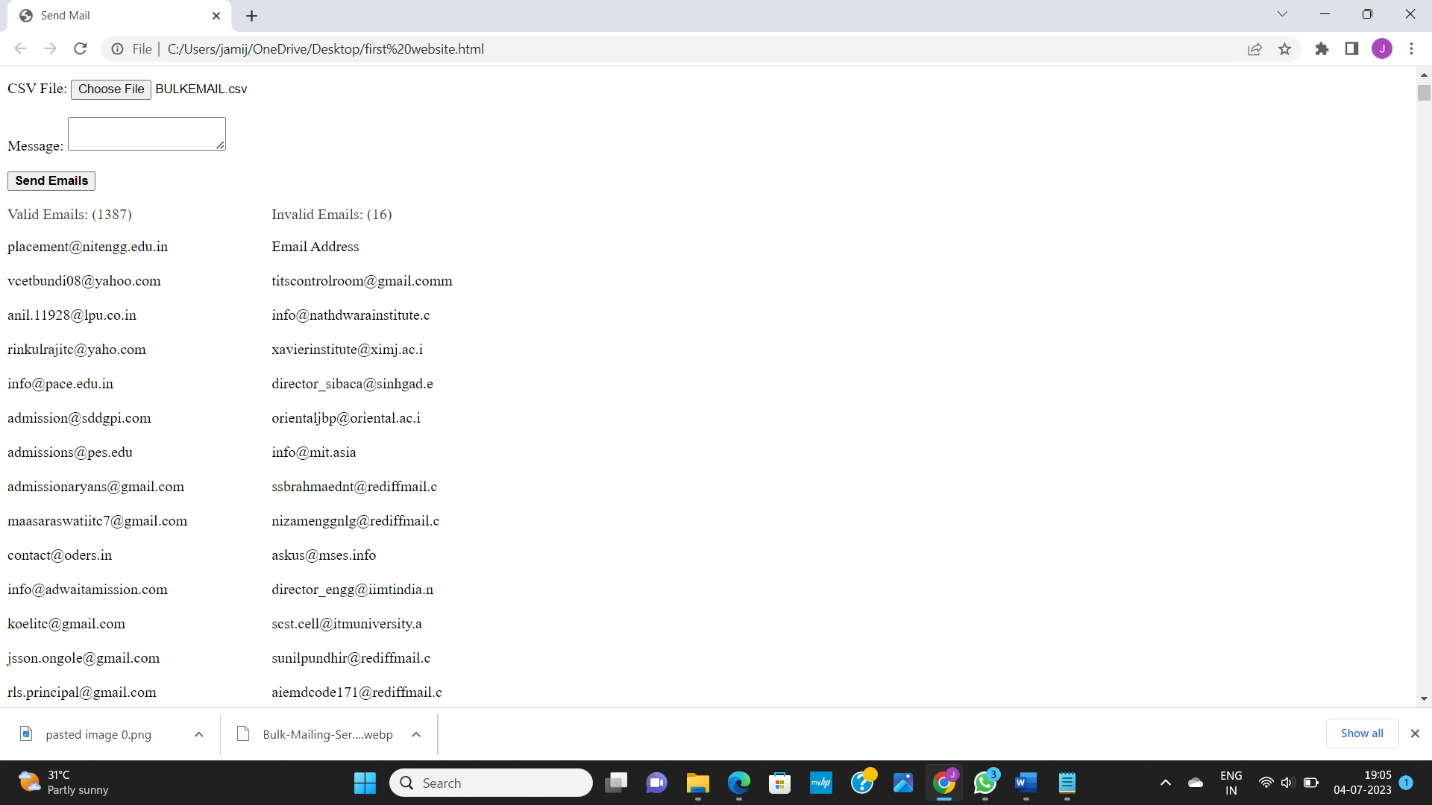
Sample output:



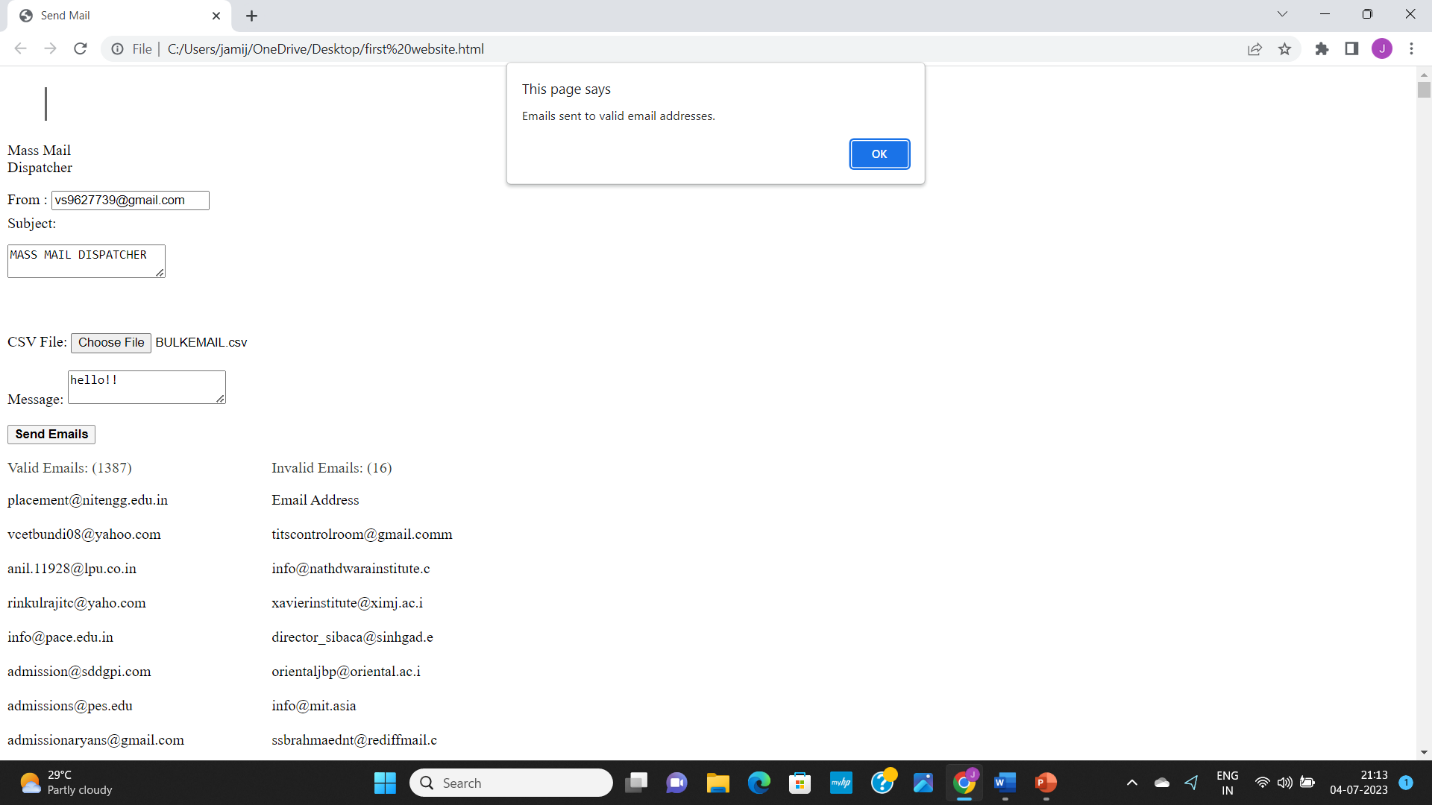
After choosing a file:



Valid||invalid emails:



Email sent to valid email:



Conclusion:

* Throughout the project, we have designed and implemented a scalable and modular architecture that includes key components such as user interface, mailing list management, email template engine, scheduler, delivery engine, and analytics and reporting.
* The system integrates with email servers or SMTP providers to ensure reliable and efficient email delivery. It incorporates security measures to protect user data and complies with email marketing regulations, such as providing opt-out options and following anti-spam practices.