**About the web app**

The web application has been successfully deployed, and you can access it using the following link. We welcome you to explore our work and provide your feedback.

[Link to the Web Application](https://mathnonesode.pythonanywhere.com/)

Feel free to navigate and evaluate our work. Your input is highly appreciated.

**About File Structure :**

**Importance :** The file structure for this **Flask web application** is critical for its proper functioning. Maintaining the file structure as described below is essential to ensure that the application runs smoothly:

File Structure for Flask Web Application:

NonESODE

│

├── app.py

├── NonESODEsolver.py

├── readme.txt

├── requirements.txt

│

├── static

│ ├── icon.jpg

│ ├── main.js

│ ├── solution.css

│ ├── solution.js

│ ├── styles.css

│ ├── venn-diagram.png

│

├── templates

│ ├── aboutOurTeam.html

│ ├── contact.html

│ ├── index.html

│ ├── solution.html

│ ├── solutiontemp.html

│ ├── userManual.html

🡪The web app can be run by executing **app.py** .Before running, install the dependency mentioned in requirments.txt (pip install -r requirements.txt).

**Note:** avoid making any changes to the file structure or filenames unless you fully understand the application, as it may lead to errors and disrupt the app's functionality.

**Non-Exact Separable ODE Solver Description**

The **non\_exact\_separable\_ode\_solver.py** -> **NonESODEsolver.py** program is designed to solve non-exact separable ordinary differential equations (ODEs) in a specific format. Here's how the equation should be structured:

* The equation should be in the form of **(M(x,y)dx + N(x,y)dy=0)**.
* Replace **M(x,y)** and **N(x,y)** with the expressions that define your ODE.
* Optionally, you can specify initial conditions for **x** and **y** . Provide these initial conditions as **x** and **y** variables in the code if needed.

The primary function, **masterSolver(eqn, x='', y='')**, takes the ODE equation and, optionally, initial conditions as input. It attempts to solve the non-exact separable ODE and provides the solution.

Other functions briefly described:

* **Solve(M, N)**: Attempts to solve the ODE by separating variables and performing integration.
* **replaceSubStr(eqn)**: Handles string substitutions for trigonometric functions and formatting.
* **reduceToStandard(eqn)**: Transforms the input equation into a standard form suitable for solving.
* **removeLog(fx, fy)**: Attempts to remove logarithmic terms from **fx** and **fy** for simplification.
* **initial\_condition(sol, x, y)**: Determines the particular solution using initial conditions, if provided.
* **expTo\_e(eqn)**: Converts "exp" to "e^" in the equation for better representation.
* **formateAnswer(solution)**: Formats the solution string for consistency.

**Example**

An example of how to use the program with the **masterSolver** function is provided in the code. You can input your non-exact separable ODE in the equation variable and, if needed, specify initial conditions for **x** and **y**.

**Conclusion**

The **Non-Exact Separable ODE Solver** is a tool for solving non-exact separable ordinary differential equations with optional initial conditions. You can manually input your ODE and find the solution. Please refer to this manual and the example code for guidance on using the program effectively.

**Please make sure to structure your non-exact separable ODE according to the provided format for successful usage.**

**Code Implementation:**

**Backend**

**NonESODEsolver.py**

import sympy as sp

from sympy.functions.elementary.trigonometric import TrigonometricFunction

import pandas as pd

import math

class NonExactODEsolver():

    nonSeparable = False

    logRemoved = False

    # validity=False

    @staticmethod

    def masterSolver(eqn: str, x='', y=''):

        try:

            solution = ''

            eqn = NonExactODEsolver.replaceSubStr(eqn)

            if not NonExactODEsolver.\_\_check\_validity(eqn):

                return "Equation is not valid"

            M, N, constant\_value = NonExactODEsolver.reduceToStandard(eqn)

            if constant\_value:

                return "This is not Ordinary Differential Equation(Constant term)"

            try:

                fx, fy = NonExactODEsolver.Solve(M, N)

                if NonExactODEsolver.nonSeparable:

                    return "EQn cannot be separated!!!"

            except:

                return "Make sure eqn is in right form!!!"

            fx\_without\_log, fy\_without\_log = NonExactODEsolver.removeLog(

                fx, fy)  # if possible

            if str(x) and str(y):

                x = int(x)

                y = int(y)

                if NonExactODEsolver.logRemoved:

                    c = NonExactODEsolver.initial\_condition(

                        fy\_without\_log/fx\_without\_log, x, y)

                    if isinstance(c, int):

                        solution = f"{NonExactODEsolver.\_\_formateAnswer(fy\_without\_log-fx\_without\_log\*c)} = 0"

                    else:

                        return "Can't get particular solution with the initial condition"

                else:

                    c = NonExactODEsolver.initial\_condition(fy-fx, x, y)

                    solution = f"{NonExactODEsolver.\_\_formateAnswer(fy)} - {NonExactODEsolver.\_\_formateAnswer(fx-c)} = 0"

            elif not str(x) and not str(y):

                if NonExactODEsolver.logRemoved:

                    solution = f"{NonExactODEsolver.\_\_formateAnswer(fy\_without\_log)} = ({NonExactODEsolver.\_\_formateAnswer(fx\_without\_log)})c"

                else:

                    solution = f"{NonExactODEsolver.\_\_formateAnswer(fy)} = {NonExactODEsolver.\_\_formateAnswer(fx)} + c"

            else:

                return "Cannot parse x,y into int."

            return solution

        except:

            return "EQn may not be in right from..Plz Check user manual..!"

    @staticmethod

    def Solve(M, N):

        x, y = sp.symbols('x y')

        M, N = NonExactODEsolver.\_\_separateVariable(M, N)

        fx = sp.integrate(M, x)

        fy = sp.integrate(N, y)

        ###################

        # remove lcm

        fx\_terms = sp.Add.make\_args(fx)  # Extract terms from the fx

        fy\_terms = sp.Add.make\_args(fy)

        numbers = []

        for term in fx\_terms+fy\_terms:

            fraction = sp.fraction(term)

            numbers.append(fraction[1])

        try:

            LCM = math.lcm(\*numbers)

            fx \*= LCM

            fy \*= LCM

        except:

            pass

        return fx, fy

    @staticmethod

    def replaceSubStr(eqn):

        '''

        This function f(x)g(y)dx + f'(x)g'(y)dy=0 to f(x)/f'(x)dx=-g'(y)/g(y)dy

        for trigonometry function:

            sine -> s

            cos -> c

            tan -> t

            cosec->S

            sec->C

            cot->T

        '''

        eqn = eqn.lower().replace(" ", "")

        eqn = eqn.replace("dx", "X")

        eqn = eqn.replace("dy", "Y")

        # for trigonometry function:

        eqn = eqn.replace("sin", "s").replace("cos", "c").replace(

            "tan", "t").replace("cosec", "S").replace("sec", "C").replace("cot", "T")

        # it will manage sinx -> sin(x) for built in function capable

        replace = {"sx": "s(x)", "cx": "c(x)", "sy": "s(y)", "cy": "c(y)", "tx": "t(x)", "ty": "t(y)",

                   "Sx": "S(x)", "Sy": "S(y)", "Cx": "C(x)", "Cy": "C(y)", "Tx": "T(x)", "Ty": "T(y)"}

        for key, value in replace.items():

            eqn = eqn.replace(key, value)

        return eqn

    @staticmethod

    def reduceToStandard(eqn: str):

        # Symbolify

        x, y, e, c, s, t, S, C, T, X, Y = sp.symbols('x y e c s t S C T X Y')

        '''

        This function f(x)g(y)dx + f'(x)g'(y)dy=0 to f(x)/f'(x)dx=-g'(y)/g(y)dy

        for trigonometry function:

            sine -> s

            cos -> c

            tan -> t

            cosec->S

            sec->C

            cot->T

            we got replaced substr eqn

        '''

        eqn = NonExactODEsolver.\_\_add\_stars(NonExactODEsolver, eqn)

        # reverse SubStr eqn

        eqn = eqn.replace("s", "sin").replace("c", "cos").replace(

            "t", "tan").replace('S', "cosec").replace("C", "sec").replace("T", "cot")

        # Making Right Hand Side 0

        Lhs = eqn[0:eqn.find("=")]

        Rhs = eqn[eqn.find("=")+1:]

        LHSEq = sp.simplify(Lhs)

        RHSEq = sp.simplify(Rhs)

        Eqn = LHSEq-RHSEq

        # handle exp function

        Eqn = Eqn.subs('e', sp.E)

        constant\_value = Eqn.subs({X: 0, Y: 0})

        M = -1 \* sp.diff(Eqn, X)  # -1 for make it rhs

        N = sp.diff(Eqn, Y)

        # Remove any fraction from M,N

        M = sp.factor(M)

        N = sp.factor(N)

        fracM = sp.fraction(M)

        fracN = sp.fraction(N)

        M = M\*fracM[1]\*fracN[1]

        N = N\*fracM[1]\*fracN[1]

        return M, N, constant\_value

    @staticmethod

    def \_\_separateVariable(M, N):

        x, y = sp.symbols("x,y")

        Mf = NonExactODEsolver.\_\_getFactor(M)

        Nf = NonExactODEsolver.\_\_getFactor(N)

        # accessing static variable

        NonExactODEsolver.nonSeparable = False

        for factor in Mf:

            # for trigonometricfunction

            if factor.has(TrigonometricFunction):

                arg1 = factor.args[0]

                if "x" in str(arg1) and "y" in str(arg1):

                    NonExactODEsolver.nonSeparable = True

            # here we replace exp to E so that our programme dose not treat (x in exp\*\*y) as true

            if 'y' in str(factor).replace('exp', 'E') and not NonExactODEsolver.nonSeparable:

                NonExactODEsolver.nonSeparable = 'y' in str(factor).replace(

                    'exp', 'E') and 'x' in str(factor).replace('exp', 'E')

                M = M/factor

                N = N/factor

        for factor in Nf:

            # for trigonometricfunction

            if factor.has(TrigonometricFunction):

                arg1 = factor.args[0]

                if "x" in str(arg1) and "y" in str(arg1):

                    NonExactODEsolver.nonSeparable = True

            if 'x' in str(factor).replace('exp', 'E') and not NonExactODEsolver.nonSeparable:

                NonExactODEsolver.nonSeparable = 'y' in str(factor).replace(

                    'exp', 'E') and 'x' in str(factor).replace('exp', 'E')

                M = M/factor

                N = N/factor

        return M, N

    @staticmethod

    def \_\_getFactor(expression):

        factors\_list = sp.factor\_list(expression)

        coefficient, terms = factors\_list

        result = []

        for term, power in terms:

            result.append(term \*\* power)

        return result

    @staticmethod

    def initial\_condition(sol, x, y):

        sol = sol.expand()

        try:

            c = int(sol.subs({'x': x, 'y': y}))

            return c

        except:

            return 'cannot get c'

###############################################################################

############### Preprocess the input text ##################################

    @staticmethod

    def \_\_check\_validity(eqn: str):

        valid\_char = "sctSCTexyXY0123456789+-\*/^()= "  # space should be counted

        return all(char in valid\_char for char in eqn) and "=" in eqn and len(eqn[eqn.find("=") + 1:])

    @staticmethod

    def \_\_add\_stars(self, input\_string):

        result = []

        prev\_char = ''

        for char in input\_string:

            if prev\_char.isalnum() and char.isalnum():

                result.append('\*')

            elif prev\_char == ')' and char.isalnum():

                result.append('\*')

            elif prev\_char == ')' and char == '(':

                result.append('\*')

            elif prev\_char == '(' and char == ')':

                result.append('\*')

            elif prev\_char.isalnum() and not (prev\_char in "sctSCT") and char == '(':

                result.append('\*')

            result.append(char)

            prev\_char = char

        return ''.join(result)

##################################################################################################

########################### Post Process Output ##################################

    @staticmethod

    def removeLog(fx, fy):

        fx\_terms = sp.Add.make\_args(fx)  # Extract terms from the fx

        fy\_terms = sp.Add.make\_args(fy)  # Extract terms from the  fy

        # accessing static variable

        NonExactODEsolver.logRemoved = False

        fx\_without\_log = 1

        fy\_without\_log = 1

        if all(['log' in str(term) for term in fx\_terms+fy\_terms]):

            NonExactODEsolver.logRemoved = True

            # for fx

            for term in fx\_terms:

                fx\_without\_log \*= sp.exp(term)

            # for fy

            for term in fy\_terms:

                fy\_without\_log \*= sp.exp(term)

            return fx\_without\_log, fy\_without\_log

        return fx, fy

    @staticmethod

    def \_\_create\_eqn\_from\_str(eqn: str):

        x, y, X, Y = sp.symbols('x y X Y')

        y = sp.Function('y')(x)

        y\_dot = y.diff(x)

        eqn = eqn.replace("dx", "X").replace("dy", "Y")

        eqn = NonExactODEsolver.\_\_add\_stars(

            NonExactODEsolver, eqn)  # add stars

        lhs = eqn[0:eqn.find("=")]

        rhs = eqn[eqn.find("=")+1:]

        Lhs = sp.simplify(lhs)

        Rhs = sp.simplify(rhs)

        expression = Lhs-Rhs

        # The eqn is not Exact or non\_Exact ODE because of constant term.

        constant\_value = expression.subs({'X': 0, 'Y': 0})

        expression = expression.subs('X', 1)

        expression = expression.subs({'y': y, 'Y': y\_dot})

        return expression, constant\_value

    @staticmethod

    def \_\_create\_Equation\_from\_roots(roots):

        y = sp.symbols('y')

        if isinstance(roots, list):

            expression = sp.Mul(\*[(y - root.rhs)

                                for root in roots], evaluate=False)

        else:

            expression = y-roots.rhs

        # eqn = sp.Eq(equation, 0)

        return sp.expand(expression)  # sp.together(eqn)

    @staticmethod

    def \_\_remove\_sqrt(equation):

        x = sp.Symbol("x")

        eq\_parts = equation.as\_ordered\_terms()

        sqrt\_term = 0

        for term in eq\_parts:

            if "sqrt" in str(term):

                sqrt\_term = term

        if sqrt\_term:

            k = 1  # sign fixer

            if sqrt\_term.subs({x: 1, "C1": 1}) < 0:  # for negative sing

                k = -1

            lhs = equation-sqrt\_term

            lhs = lhs\*lhs

            sqrt\_term = sqrt\_term\*sqrt\_term

            equation = lhs+sqrt\_term\*k

        return equation

    @staticmethod

    def expTo\_e(eqn):

        terms = sp.Add.make\_args(eqn)

        sum = 0

        for i in range(len(terms)):

            factor\_list = sp.factor\_list(terms[i])

            a = 1

            for j in range(len(factor\_list)):

                if "exp" in str(factor\_list[j]):

                    factor\_list[j] = sp.log(factor\_list[j])

                a \*= factor\_list[j]

            sum += terms[i]

        return sum

    @staticmethod

    def \_\_formateAnswer(solution):

        solution = solution.cancel()

        solution = solution.factor()

        solution = solution.simplify()

        eqn = str(solution)

        eqn = eqn.replace("\*\*", "^").replace("\*", "").replace("C1",

                                                              "c").replace("c^2", 'c').replace("exp", "e^")

        return eqn

    #######################################################################################################

    ###################### For Exact Ordinary Differential Equation#################################

    def \_\_ExactODE(self, m, n):

        x, y, c = sp.symbols('x y c')

        # Define the coefficients M(x, y) and N(x, y)

        M = sp.simplify(m)  # Define your M(x, y) function here

        N = sp.simplify(n)   # Define your N(x, y) function here

        # Calculate partial derivatives

        M\_y = sp.diff(M, y)

        N\_x = sp.diff(N, x)

        # Check if the equation is exact

        if M\_y == N\_x:

            # If it's exact, find the potential function F(x, y)

            F = sp.integrate(M, x)  # + f(y)

            Fy = sp.diff(F, y)  # + f'(y)

            # f'(y)=N-Fy

            Y = sp.integrate(N-Fy)

            solve = F+Y

            ans = str(solve) + " = c"

            return NonExactODEsolver.formateAnswer(ans)

        else:

            return "dM/dy != dN/dx \nNot Exact ODE \nUnable to Solve"

##################### END class ####################################################################################################

def patternExtractor(input):

    import re

    # Input string

    input\_string = "y(a)=b"

    # Define a regular expression pattern to match 'a' and 'b'

    pattern = r'y\((.\*?)\)=(.\*)'

    # Use re.search to find the pattern in the input string

    match = re.search(pattern, input\_string)

    # Check if a match was found

    if match:

        a = match.group(1)  # Extract the value of 'a'

        b = match.group(2)  # Extract the value of 'b'

        return a, b

    else:

        False, False

def CSV\_solve(file):

    df = pd.read\_csv(file)

    ans = ''

    for eqn in df['equation']:

        print(eqn)

        ans = NonExactODEsolver.masterSolver(eqn)

        print(sol)

    return

    ans = df['equation'].apply(NonExactODEsolver.masterSolver)

    # ans=NonExactODEsolver.Solve2(df['equation'][0])

    df.insert(2, "solution\_by\_programme", ans)

    df.to\_csv("SolveFF.csv", index=False)

**Frontend**

**templates/Index.html**

<!DOCTYPE html>

<html lang="en">

<head>

   <title>NonExact separable ODE</title>

   <meta charset="UTF-8">

   <meta http-equiv="X-UA-Compatible" content="IE=edge">

   <meta name="viewport" content="width=device-width, initial-scale=1.0">

   <!--  -->

   <link rel="stylesheet" href="../static/styles.css">

   <script src="../static/main.js"></script>

   <link rel="icon" type="image/x-icon" href="../static/icon.jpg">

   <!--  -->

   <link rel="preconnect" href="https://fonts.googleapis.com">

   <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>

   <link

      href="https://fonts.googleapis.com/css2?family=Inclusive+Sans&family=Mooli&family=Roboto:ital,wght@0,300;0,400;1,300&display=swap"

      rel="stylesheet">

   <!--  -->

</head>

<body>

   <div class="welcomediv" id="welcomediv">

      <!-- Navigation section -->

      <div id="mySidenav" class="sidenav">

         <a href="javascript:void(0)" class="closebtn" onclick="closeNav()">&times;</a>

         <a href="/userManual" onclick="closeNav()" target="\_blank">User Manual</a>

         <a href="/aboutOurTeam" onclick="closeNav()">About Our Team</a>

         <a href="/contact" onclick="closeNav()">Contact</a>

      </div>

      <!-- END Navigation section -->

      <!-- Navigation button -->

      <span class="navbar" style="font-size:30px;cursor:pointer" onclick="openNav()">&#9776;</span>

      <br>

      <br>

      <div>

         <p>

            <h1 class="h11">Welcome To Our Program</h1>

            <br>

            <br>

            <h1 class="h22">Differential Equation Solver</h1>

         </p>

         <p class="intro">

         <button id="start" class="buttonDeafult" type="button" onclick="toggleDivs()">Click To Start</button>

         </p>

      </div>

   </div>

   <div class="solutionDiv" id="solutionDiv">

      <button class="cross" title="Close" type="button" onclick="backtostart();solveAnother();f2();f1();">X</button>

      <p>

      <form class="solveform" id="solveforms"  method="post">

         <label id="lbl" for="eqnField">Enter seperable O.D.E function :</label>

         <br><br>

         <input type="text" id="eqnField" name="eqnField" placeholder="E.g:  M(x,y)dx + N(x,y)dy = 0" required autofocus>

         <br>

         <button class="buttonDeafult" id="cnd" onclick="handleButtonClick(event)">With Initial Condition</button>

         <br>

         <div id="xy">

            <label class="xLabel" for="x">X :</label>

            <input type="number" id="x" name="x" placeholder="x">

            <label class="yLabel" for="y">Y :</label>

            <input type="number" id="y" name="y" placeholder="y">

         </div>

         <button class="buttonDeafult" type="button" id="resetForm" onclick="solveAnother()">Clear Equation</button>

         <button class="buttonDeafult" value="submit" id="solve" type="submit">Solve</button>

      </form>

      <!-- <br><br> -->

   </div>

   <!-- waiting div -->

   <div class="waitingdiv" id="waitingdiv">

      <p class="div3p1">Our program is awaiting your confirmation</p>

      <p class="div3p2" >

         Click the button to proceed

         <br>

         <div class="container">

         <div class="dot"></div>

         <div class="dot"></div>

         <div class="dot"></div>

         </div>

      </p>

   </div>

   <!-- for solve Another ..it prevent waiting div to show up -->

   <script>

      // Get the function name from the query parameter

      var functionName = '{{ toggleDivs }}';

      // Call the function if it exists

      if (functionName && typeof window[functionName] === 'function') {

         window[functionName]();

      }

  </script>

</body>

</html>

**templates/solution.html**

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Non Exact separable ODE</title>

    <link rel="icon" type="image/x-icon" href="../static/icon.jpg">

    <!--  -->

    <link rel="preconnect" href="https://fonts.googleapis.com">

    <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>

    <link

        href="https://fonts.googleapis.com/css2?family=Inclusive+Sans&family=Mooli&family=Roboto:ital,wght@0,300;0,400;1,300&display=swap"

        rel="stylesheet">

    <!--  -->

    <link rel="stylesheet" href="../static/solution.css">

</head>

<body>

<div class="header">

        <h1>NonExact separable ODE</h1>

    </div>

    <div class="solution-container">

        <div class="boxed-content">

            <h3>Entered Equation</h3>

            <p>

                    {% if eqn %} {{eqn}} {%endif%}

                </p>

        </div>

        <div class="boxed-content">

            <h3>Solution</h3>

            <p>

                    {% if solution %}{{solution}}{%endif%}</p>

        </div>

    </div>

    <form style="text-align: center" method="post">

        <button class="solveAnother" id="solveAnother" type="submit">

            Solve Another

        </button>

    </form>

</body>

</html>

**templates/userManual.html**

<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="UTF-8" />

    <meta http-equiv="X-UA-Compatible" content="IE=edge" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <title>About</title>

    <link rel="icon" type="image/x-icon" href="../static/venn-diagram.png" />

    <link rel="preconnect" href="https://fonts.googleapis.com" />

    <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin />

    <link

      href="https://fonts.googleapis.com/css2?family=Inclusive+Sans&family=Mooli&family=Roboto:ital,wght@0,300;0,400;1,300&display=swap"

      rel="stylesheet"

    />

    <style>

      \* {

        font-family: "Mooli", sans-serif;

      }

      table,

      th,

      td {

        border: 5px solid bisque;

        border-collapse: collapse;

        background-color: black;

      }

      tr {

        height: 60px;

      }

      tr:hover {

        color: deepskyblue;

      }

      th {

        font-size: 28px;

      }

      td {

        font-size: 21px;

      }

      table {

        margin-bottom: 20px;

      }

      caption {

        color: white;

        font-size: 40px;

        margin-top: 30px;

        margin-bottom: 40px;

        text-align: center;

      }

      body {

        background-color: rgba(0, 0, 0, 0.8);

        color: bisque;

        text-align: center;

      }

      #btn {

        font-size: 14px;

        background-color: #d9c8f0;

        padding: 10px;

        margin-top: 5px;

        border-radius: 12px;

        font-style: bold;

        width: 120px;

        height: 45px;

      }

      #btndiv {

        margin-top: 3%;

        margin-left: 5%;

      }

    </style>

  </head>

  <body>

    <div id="btndiv" style="float: left">

      <a href="/"><button id="btn">Back To Home</button></a>

    </div>

    <div style="float: left">

      <table id="main" style="margin-left: 27%; width: 90%" ;>

        <caption>

          <b>User Manual</b>

        </caption>

        <tr>

          <th style="width: 44%">Equation</th>

          <th style="width: 44%">General Solution</th>

        </tr>

        <tr>

          <td>e^ydy=(x^2-2x)dx</td>

          <td>3e^(y) = x^2(x - 3) + c</td>

        </tr>

        <tr>

          <td>dy/(y+1)=dx/(x-1)</td>

          <td>y + 1 = (x - 1)c</td>

        </tr>

        <tr>

          <td>ydy/(1+y^2)=dx/(x(1+x^2))</td>

          <td>y^2 + 1 = (x^2/(x^2 + 1))c</td>

        </tr>

        <tr>

          <td>dy/(e^-y)=(e^x+x^2)dx</td>

          <td>3e^(y) = x^3 + 3e^(x) + c</td>

        </tr>

        <tr>

          <td>((x-4)/x^3)dx=((y^2-3)/y^4)dy</td>

          <td>(y^2 - 1)/y^3 = (x - 2)/x^2 + c</td>

        </tr>

        <tr>

          <td>ydy=-xe^-(x^2)dx</td>

          <td>y^2/2 = e^(-x^2)/2 + c</td>

        </tr>

        <tr>

          <td>(1-y^2)dy=x^2dx</td>

          <td>y(3 - y^2) = x^3 + c</td>

        </tr>

        <tr>

          <td>2(y-1)dy=(3x^2+4x+2)dx</td>

          <td>y(y - 2) = x(x^2 + 2x + 2) + c</td>

        </tr>

        <tr>

          <td>((1-2y^2)/y)dy=cos(x)dx</td>

          <td>-y^2 + log(y) = sin(x) + c</td>

        </tr>

        <tr>

          <td>dy/(1+y^2)=dx/(1+x^2)</td>

          <td>atan(y) = atan(x) + c</td>

        </tr>

        <tr>

          <td>(cos(x)/sin(x))dx=-(e^y/(e^y+1))dy</td>

          <td>e^(y) + 1 = (1/sin(x))c</td>

        </tr>

      </table>

    </div>

    <script>

      document.getElementById("btn").addEventListener("click", function () {

        // Close the current tab or window

        window.close();

      });

    </script>

  </body>

</html>

**templates/aboutOurTeam.html**

<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8" />

  <meta http-equiv="X-UA-Compatible" content="IE=edge" />

  <meta name="viewport" content="width=device-width, initial-scale=1.0" />

  <title>About</title>

  <link rel="icon" type="image/x-icon" href="../static/icon.jpg">

  <link rel="preconnect" href="https://fonts.googleapis.com">

  <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>

  <link

    href="https://fonts.googleapis.com/css2?family=Inclusive+Sans&family=Mooli&family=Roboto:ital,wght@0,300;0,400;1,300&display=swap"

    rel="stylesheet">

  <style>

    \* {

      font-family: "Mooli", sans-serif;

    }

    table,

    th,

    td {

      border: 5px solid bisque;

      border-collapse: collapse;

      background-color: black;

    }

    tr {

      height: 60px;

    }

    tr:hover {

      color: deepskyblue;

    }

    th {

      font-size: 28px;

    }

    td {

      font-size: 24px;

    }

    table {

      margin-bottom: 20px;

    }

    caption {

      color: white;

      font-size: 40px;

      margin-top: 30px;

      margin-bottom: 40px;

      text-align: center;

    }

    body {

      background-color: rgba(0, 0, 0, 0.8);

      color: bisque;

      text-align: center;

    }

    #solveAnother {

      font-size: 14px;

      background-color: #d9c8f0;

      padding: 10;

      margin-top: 5;

      border-radius: 12px;

      /\* font-family: Verdana; \*/

      font-style: bold;

      width: 120px;

      height: 45px;

      margin-top: 5%;

      margin-left: 5%;

    }

    </style>

</head>

<body>

  <a href="/">

    <button type="button" style="float: left;" id="solveAnother">Back To Home</button>

  </a>

  <table id="main" style="margin-left: 15%; width: 70%" ;>

    <caption><b>About Our Team</b></caption>

    <tr>

      <th style="width:42%">Name</th>

      <th style="width:20%">Roll</th>

      <th style="width:42%">Contribution</th>

    </tr>

    <tr>

      <td>Nazrul Islam Ridwan</td>

      <td>2103164</td>

      <td>Website Developer</td>

    </tr>

    <tr>

      <td>Hirok Reza</td>

      <td>2103153</td>

      <td>Programmer</td>

    </tr>

    <tr>

      <td>Rakib Hossain</td>

      <td>2103172</td>

      <td>Math Solver</td>

    </tr>

    <tr>

      <td>Sagar Mollah</td>

      <td>2103176</td>

      <td>Math Solver</td>

    </tr>

    <tr>

      <td>Rimon Chakma</td>

      <td>2103181</td>

      <td>Math Solver</td>

    </tr>

    <tr>

      <td>Bayezid Hasan</td>

      <td>2103170</td>

      <td>Math Solver</td>

    </tr>

    <tr>

      <td>Shakibur Rahman</td>

      <td>2103177</td>

      <td>Math Solver</td>

    </tr>

    <tr>

      <td>Imamul Haque</td>

      <td>2103145</td>

      <td>Math Solver</td>

    </tr>

    <tr>

      <td>Isteak Shajal</td>

      <td>2103138</td>

      <td>Presenter</td>

    </tr>

    <tr>

      <td>Limon Hasan</td>

      <td>2103167</td>

      <td>Presenter</td>

    </tr>

    <tr>

      <td>Maruf Billah</td>

      <td>2103156</td>

      <td>Co-ordinator</td>

    </tr>

  </table>

  <script>

    document.getElementById("solveAnother").addEventListener("click", function () {

      // Close the current tab or window

      window.close();

    });

  </script>

</body>

</html>

**templates/contact.html**

<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="UTF-8" />

    <meta http-equiv="X-UA-Compatible" content="IE=edge" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <title>Contact</title>

    <link rel="icon" type="image/x-icon" href="../static/icon.jpg">

    <link rel="preconnect" href="https://fonts.googleapis.com">

    <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>

    <link href="https://fonts.googleapis.com/css2?family=Inclusive+Sans&family=Mooli&family=Roboto:ital,wght@0,300;0,400;1,300&display=swap" rel="stylesheet">

  </head>

  <style>

    \*{

   font-family: "Mooli", sans-serif;

}

    body {

      background-color: rgba(0, 0, 0, 0.8);

      color: bisque;

      font-size: 24px;

    }

    div{

      margin-left:5%;

      margin-top:4%;

    }

    #solveAnother{

font-size: 14px;

background-color: #d9c8f0;

padding: 10;

margin-top: 5;

border-radius: 12px;

font-style: bold;

width:120px;

height:48px;

margin-top:-3%;

margin-left: 5%;

}

  </style>

  <body>

   <h2 style="text-align: center;">Contact</h2>

   <a href="/"><button style="float:left; position:absolute;" id="solveAnother">Back To Home</button></a>

    <div style="margin-left: 37%;">

      <p>

        Nazrul Islam Ridwan<br />

        Email: riidwaan5060@gmail.com<br />

        Phone: 01965632132

   </p><br>

      <p>

        Hirok Reza<br />

        Email: hirokreza121@gmail.com<br />

        Phone: 01965632132

      </p><br>

      <p>

        Rakib Hossain<br />

        Email: rakib.hosen.tamim.2k2@gmail.com<br />

        Phone: 01965632132

      </p><br>

      <p>

        Sagar Mollah<br />

        Email: tahasanhabib83@gmail.com<br />

        Phone: 01965632132

      </p><br>

      <p>

        Rimon Chakma<br />

        Email: rimonno2021@gmail.com<br />

        Phone: 01965632132

      </p><br>

      <p>

        Bayezid Hasan<br />

        Email: nayezidhassan2003@gmail.com<br />

        Phone: 01965632132

      </p><br>

      <p>

        Shakibur Rahman<br />

        Email: mdshakiburr42@gmail.com<br />

        Phone: 01965632132

      </p><br>

      <p>

        Imamul Haque<br />

        Email: itk21st@gmail.com<br />

        Phone: 01965632132

      </p><br>

      <p>

        Isteak Shajal<br />

        Email: isteak.0023@gmail.com<br />

        Phone: 01965632132

      </p><br>

      <p>

        Limon Hasan<br />

        Email: limonhasan121999@gmail.com<br />

        Phone: 01965632132

      </p><br>

      <p>

        Maruf Billah<br />

        Email: marufbillah2929@gmail.com<br />

        Phone: 01965632132

      </p><br>

    </div>

    <script>

      document.getElementById("solveAnother").addEventListener("click", function() {

          // Close the current tab or window

          window.close();

      });

  </script>

  </body>

</html>

**static/main.js**

// document.addEventListener("DOMContentLoaded", function () {

  var y = 3;

  var toggleDivActive = 1;

  // For navigation

  function openNav() {

    document.getElementById("mySidenav").style.width = "250px";

  }

  function closeNav() {

    document.getElementById("mySidenav").style.width = "0";

  }

  function handleButtonClick(event) {

    event.preventDefault();

    f1();

  }

  function f1() {

    if (y === 2) {

      document.getElementById("xy").style.visibility = "hidden";

      document.getElementById("cnd").style.backgroundColor = "#d9c8f0";

      clearInput();

      y = 3;

    } else {

      document.getElementById("xy").style.visibility = "visible";

      document.getElementById("cnd").style.backgroundColor = "#b68deb";

      clearInput();

      y = 2;

    }

  }

  function clearInput() {

    document.getElementById("x").value = "";

    document.getElementById("y").value = "";

  }

  function f2() {

    y = 2;

  }

  //==========================================================

  function solveAnother() {

    document.getElementById("solveforms").reset();

  }

  function toggleDivs() {

    if (toggleDivActive) {

      var waitingdiv = document.getElementById("waitingdiv");

      var solutionDiv = document.getElementById("solutionDiv");

      var eqnField = document.getElementById("eqnField");

      eqnField.setAttribute.autofocus=true

      if (waitingdiv.style.display !== "none") {

        waitingdiv.style.display = "none";

        solutionDiv.style.display = "block";

      }

    }

  }

  function backtostart() {

    var waitingdiv = document.getElementById("waitingdiv");

    var solutionDiv = document.getElementById("solutionDiv");

    if (solutionDiv.style.display !== "none") {

      waitingdiv.style.display = "block";

      solutionDiv.style.display = "none";

    }

  }

**static/style.css**

\* {

    margin: 0;

    padding: 0;

    box-sizing: border-box;

    font-family: "Mooli", sans-serif;

}

input:placeholder-shown {

    font-style: italic;

}

.welcomediv {

    width: 50%;

    background-color: black;

    color: bisque;

    min-height: 100vh;

    text-align: center;

    float: left;

    max-height: 100vh;

}

.h11 {

    font-size: 35px;

}

.h22 {

    font-size: 27px;

}

p {

    padding: 50px;

}

.solutionDiv {

    width: 50%;

    background-color: rgba(0, 0, 0, 0.8);

    color: bisque;

    min-height: 100vh;

    text-align: center;

    float: left;

    font-size: 20px;

    max-height: 100vh;

    display: none;

}

.waitingdiv {

    width: 50%;

    background-color: rgba(0, 0, 0, 0.8);

    color: bisque;

    min-height: 100vh;

    text-align: left;

    float: left;

    font-size: 25px;

}

.div3p1 {

    margin-top: 53px;

}

.div3p2 {

    margin-top: 10px;

}

.solveform {

    margin-top: 30px;

    font-size: 25px;

}

.cross {

    font-size: 18px;

    text-align: center;

    background-color: rgb(217, 200, 240);

    padding: 2px;

    border: 2px solid white;

    border-radius: 4px;

    font-family: Verdana;

    width: 38px;

    height: 38px;

    float: right;

    margin-right: 15px;

    margin-top: 15px;

}

.cross:hover {

    background-color: rgba(216, 52, 52, 0.9);

    color: bisque;

}

/\* form //////////////////////////////////////////////\*/

#eqnField {

    margin-top: 2px;

    width: 60%;

    height: 45px;

    font-size: 20px;

}

#x,

#y {

    margin-top: 20px;

    width: 80px;

    height: 34px;

    font-size: 18px;

}

.xLabel,

.yLabel {

    font-size: 20px;

}

.xLabel {

    margin-left: -10px;

}

/\* button ///////////////////////////////////\*/

.buttonDeafult {

    font-size: 14px;

    background-color: #d9c8f0;

    padding: 13px;

    margin: 20px 20px 20px 20px;

    /\* border: 2px solid black; \*/

    border-radius: 9px;

    font-family: Verdana;

    font-style: bold;

    transition: background-color 0.3s ease;

}

.buttonDeafult:hover {

    background-color: #b68deb;

}

#solve {

    width: 100px;

}

/\* nav bar /////////////////////////////////////////\*/

.sidenav {

    height: 100%;

    width: 0;

    position: fixed;

    z-index: 1;

    top: 0;

    left: 0;

    background-color: #111;

    overflow-x: hidden;

    transition: 0.5s;

    padding-top: 60px;

}

.sidenav a {

    padding: 8px 8px 8px 32px;

    text-decoration: none;

    font-size: 25px;

    color: #818181;

    display: block;

    transition: 0.3s;

}

.sidenav a:hover {

    color: #f1f1f1;

}

.sidenav .closebtn {

    position: absolute;

    top: 0;

    right: 25px;

    font-size: 36px;

    margin-left: 50px;

}

@media screen and (max-height: 450px) {

    .sidenav {

        padding-top: 15px;

    }

    .sidenav a {

        font-size: 18px;

    }

}

.navbar {

    float: left;

    margin-left: 20px;

    margin-top: 16px;

    display: block;

}

/\* dots////////////////////////////////////////////////// \*/

.dot {

    display: inline-block;

    width: 12.5px;

    height: 12.5px;

    border-radius: 50%;

    background-color: #4b9cdb;

}

/\* #4b9cdb \*/

@keyframes jumpingAnimation {

    0% {

        transform: translate3d(0, 0, 0);

    }

    50% {

        transform: translate3d(0, 15px, 0);

    }

    100% {

        transform: translate3d(0, 0, 0);

    }

}

.container {

    margin-left: 55px;

    margin-top: -30px;

}

.container .dot:nth-last-child(1) {

    animation: jumpingAnimation 0.6s 0.1s ease-in infinite;

}

.container .dot:nth-last-child(2) {

    animation: jumpingAnimation 0.6s 0.2s ease-in infinite;

}

.container .dot:nth-last-child(3) {

    animation: jumpingAnimation 0.6s 0.3s ease-in infinite;

}

.hidden2 {

    display: none;

}

#div4 {

    float: right;

    margin-top: -20px;

    padding-right: 120px;

}

#xy {

    visibility: hidden;

    align-items: center;

    margin-top: -18px;

}

#start {

    margin-top: -250px;

}

#solveforms {

    margin-top: 13%;

}

#lbl {

    font-size: 125%;

}

/\* //////////////////////////////////////////////// \*/

#myModal {

  display: none;

  position: fixed;

  z-index: 1;

  left: 25%; /\* Adjust as needed \*/

  top: 10%;

  width: 50%;

  height: 80%;

  overflow: auto;

  background-color: lightslategray;

}

#modalContent {

  background-color: transparent;

  margin: 0;

  padding: 0;

  border: none;

  width: 100%;

  height: 100%;

  box-shadow: none;

  position: relative;

}

/\* Close button style \*/

#closeButton {

  position: absolute;

  top: 0;

  right:0;

  padding: 10px;

  cursor: pointer;

}

#modalIframe {

  width: 100%;

  height: 100%;

  border: none; /\* Remove iframe border \*/

}

**static/solution.css**

body {

  font-family: "Mooli", sans-serif;

  background-color: rgba(0, 0, 0, 0.8);

  color: azure;

  margin: 0;

  padding: 0;

}

.header {

  text-align: center;

  padding: 50px 0;

  width: 100%;

}

.header h1 {

  font-size: 36px;

}

.solution-container {

  text-align: center;

  padding: 20px;

  margin: 20px;

  display: flex;

  flex-direction: column;

  align-items: center;

}

.boxed-content {

  max-width: 600px;

  width: 100%;

  /\* Make sure the content takes the full width within the container \*/

  vertical-align: top;

}

.boxed-content h3 {

  font-size: 24px;

  margin: 10px 0;

}

.boxed-content p {

  font-size: 18px;

  margin: 10px 0;

  border: 2px solid #0078d4;

  border-radius: 10px;

  padding: 10px;

  margin: 10px 0;

  background-color: dodgerblue;

}

.solveAnother {

  display: inline-block;

  padding: 10px 20px;

  font-size: 20px;

  background-color: #d9c8f0;

  /\* color: white; \*/

  text-decoration: none;

  border-radius: 5px;

  font-family: Verdana;

  font-style: bold;

  transition: background-color 0.3s ease;

}

.solveAnother:hover {

  background-color: #b68deb;

}

**API side**

**app.py**

from flask import Flask, request, render\_template,redirect,url\_for,session

from NonESODEsolver import NonExactODEsolver

app = Flask(\_\_name\_\_)

app.secret\_key="secret\_key"

solveAnother=False

@app.route('/', methods=['GET', 'POST'])

def index():

    result=''

    global solveAnother

    if request.method == 'POST':

        eqn = request.form.get('eqnField')

        session["eqn"]=eqn

        x = request.form.get('x')

        y = request.form.get('y')

        if x and y:

            result = NonExactODEsolver.masterSolver(eqn, int(x), int(y))

        else:

            result = NonExactODEsolver.masterSolver(eqn)

        session["sol"]=result

        return redirect(url\_for("solution"))

    if solveAnother:

        solveAnother=False

        return render\_template('index.html',toggleDivs="toggleDivs")

    else:

        return render\_template('index.html')

@app.route('/solution', methods=['GET', 'POST'])

def solution():

    eqn=session.get("eqn")

    sol=session.get("sol")

    if request.method == 'POST':

        global solveAnother

        solveAnother=True

        return redirect('/')

    return render\_template('solution.html',eqn=eqn,solution=sol)

@app.route('/aboutOurTeam')

def aboutOurTeam():

    return render\_template('aboutOurTeam.html')

@app.route('/contact')

def contact():

    return render\_template('contact.html')

@app.route('/userManual')

def userManual():

    return render\_template('userManual.html')

if \_\_name\_\_ == '\_\_main\_\_':

    app.run(debug=True)