SPIRNT-4

TEAM ID	PNT2022TMID17480
Project Title	Early Detection Of Chronic Kidney Disease Using Machine Learning

```
{
"cells": [
 {
  "cell_type": "code",
  "execution_count": 1,
  "metadata": {},
  "outputs": [],
  "source": [
   "import pickle\n",
   "loaded_class = pickle. load(open('randomclass_chronic', 'rb'))\n",
   "loaded_reg = pickle. load(open('randomreg_chronic', 'rb'))"
  ]
 },
 {
   "cell_type": "code",
   "execution_count": 2,
   "metadata": {},
   "outputs": [],
   "source": [
    "import numpy as np\n"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 3,
   "metadata": {},
   "outputs": [],
   "source": [
    "import pandas as pd"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "metadata": {},
   "outputs": [
    {
     "name": "stdout",
     "output_type": "stream",
     "text": [
      " * Serving Flask app \"__main__\" (lazy loading)\n",
      " * Environment: production\n",
      "   WARNING: This is a development server. Do not use it in a production deployment.\n",
      "   Use a production WSGI server instead.\n",
      " * Debug mode: off\n"
     ]
    },
    {
     "name": "stderr",
     "output_type": "stream",
     "text": [
      " * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)\n",
      "127.0.0.1 - - [03/Nov/2022 00:07:59] \"\u001b[37mGET / HTTP/1.1\u001b[0m\" 200 -\n",
      "C:\\Users\\Sinegalatha\\anaconda3\\lib\\site-packages\\sklearn\\base.py:450: UserWarning: X does not
```

```
have valid feature names, but RandomForestClassifier was fitted with feature names\n",
      "  warnings.warn(\n",
      "C:\\Users\\Sinegalatha\\anaconda3\\lib\\site-packages\\sklearn\\base.py:450: UserWarning: X does not have valid feature names, but RandomForestRegressor was fitted with feature names\n",
      "  warnings.warn(\n",
      "127.0.0.1 - - [03/Nov/2022 00:09:40] \"\u001b[37mPOST /val HTTP/1.1\u001b[0m\" 200 -\n"
     ]
    },
    {
     "name": "stdout",
     "output_type": "stream",
     "text": [
      "['48', '70', '1', '4', '0', 0, 1, 1, 0, '117', '56', '4', '111', '3', '11', '32', '6700', '4', 1, 0, 0, 0, 1, 1]\n"
     ]
    },
    {
     "name": "stderr",
     "output_type": "stream",
    "text": [
     "127.0.0.1 - - [03/Nov/2022 00:09:47] \"\u001b[37mGET / HTTP/1.1\u001b[0m\" 200 -\n",
     "C:\\Users\\Sinegalatha\\anaconda3\\lib\\site-packages\\sklearn\\base.py:450: UserWarning: X does not have valid feature names, but RandomForestClassifier was fitted with feature names\n",
     "  warnings.warn(\n",
     "C:\\Users\\Sinegalatha\\anaconda3\\lib\\site-packages\\sklearn\\base.py:450: UserWarning: X does not have valid feature names, but RandomForestRegressor was fitted with feature names\n",
     "  warnings.warn(\n",
     "127.0.0.1 - - [03/Nov/2022 00:11:12] \"\u001b[37mPOST /val HTTP/1.1\u001b[0m\" 200 -\n"
    ]
   },
   {
    "name": "stdout",
    "output_type": "stream",
    "text": [
     "['58', '80', '1', '0', '0', 0, 0, 0, 0, '131', '18', '1', '141', '4', '16', '53', '6800', '6', 0, 0, 0, 1, 0, 0]\n"
    ]
    }
   ],
   "source": [
    "from flask import Flask, request, redirect, render_template\n",
    "app = Flask(__name__)\n",
    "@app.route(\"/\",methods=['GET', 'POST'])\n",
    "def index():\n",
    "    return render_template('index.html')\n",
    "@app.route(\"/val\",methods=['POST'])\n",
    "\n",
    "def val():\n",
    "    test=[]\n",
    "    if request.method == 'POST':\n",
    "        test.append(request.form.get(\"age\"))\n",
    "        test.append(request.form.get(\"bp\"))\n",
    "        test.append(request.form.get(\"sg\"))\n",
    "        test.append(request.form.get(\"al\"))\n",
    "        test.append(request.form.get(\"su\"))\n",
    "        rb=request.form.get(\"rbc\")\n",
    "        if rb=='abnormal':\n",
    "            test.append(1)\n",
    "    else:\n",
    "        test.append(0)\n",
    "    pc=request.form.get(\"pc\")\n",
    "    if pc=='abnormal':\n",
    "        test.append(1)\n",
    "    else:\n",
    "        test.append(0)\n",
    "    pcc=request.form.get(\"pcc\")\n",
    "    if pcc=='present':\n",
    "        test.append(1)\n",
    "    else:\n",
    "        test.append(0)\n",
    "    ba=request.form.get(\"ba\")\n",
    "    if ba=='present':\n",
    "        test.append(1)\n",
    "    else:\n",
    "        test.append(0)\n",
    "    test.append(request.form.get(\"bgr\"))\n",
    "    test.append(request.form.get(\"bu\"))\n",
    "    test.append(request.form.get(\"sc\"))\n",
    "    test.append(request.form.get(\"sod\"))\n",
    "    test.append(request.form.get(\"pot\"))\n",
    "       test.append(request.form.get(\"hemo\"))\n",
    "       test.append(request.form.get(\"pcv\"))\n",
    "       test.append(request.form.get(\"wc\"))\n",
    "       test.append(request.form.get(\"rc\"))\n",
    "       ht=request.form.get(\"htn\")\n",
    "       if ht=='yes':\n",
    "           test.append(1)\n",
    "       else:\n",
    "           test.append(0)\n",
    "       d=request.form.get(\"dm\")\n",
    "       if d=='yes':\n",
    "           test.append(1)\n",
    "       else:\n",
    "           test.append(0)\n",
    "       ca=request.form.get(\"cad\")\n",
    "       if ca=='yes':\n",
    "           test.append(1)\n",
    "       else:\n",
    "           test.append(0)\n",
    "       ap=request.form.get(\"appet\")\n",
    "       if ap=='good':\n",
    "           test.append(1)\n",
    "        elif ap=='poor':\n",
    "            test.append(0)\n",
    "        else:\n",
    "            test.append(np.nan)\n",
    "        p=request.form.get(\"pe\")\n",
    "        if p=='yes':\n",
    "            test.append(1)\n",
    "        else:\n",
    "            test.append(0)\n",
    "        an=request.form.get(\"ane\")\n",
    "        if an=='yes':\n",
    "            test.append(1)\n",
    "        else:\n",
    "            test.append(0)\n",
    "    print(test)\n",
    "    test_df=pd.DataFrame(test)\n",
    "    test_df=np.array(test_df).reshape(1, -1)\n",
    "    \n",
    "    ans1=loaded_class.predict(test_df)\n",
    "    ans2=loaded_reg.predict(test_df)\n",
    "    if int(ans1)==1:\n",
        "        answer1=\"Sorry to say!! You have CHRONIC DISEASE!!!\"\n",
        "        return render_template('rename.html',answer1=answer1,answer2=ans2)\n",
        "    else:\n",
        "        answer1=\"Happy to say that you don't have CHRONIC DISEASE\"\n",
        "\n",
        "        return render_template('rename2.html',answer1=answer1,answer2=ans2)\n",
        "    \n",
        "if __name__ == \"__main__\":\n",
        "    app.debug=True\n",
        "    app.run(debug=False)"
       ]
      },
      {
       "cell_type": "code",
       "execution_count": null,
       "metadata": {},
       "outputs": [],
       "source": []
      },
      {
       "cell_type": "code",
       "execution_count": null,
       "metadata": {},
       "outputs": [],
       "source": []
      }
     ],
     "metadata": {
      "kernelspec": {
       "display_name": "Python 3",
       "language": "python",
       "name": "python3"
      },
      "language_info": {
       "codemirror_mode": {
        "name": "ipython",
        "version": 3
       },
       "file_extension": ".py",
       "mimetype": "text/x-python",
       "name": "python",
       "nbconvert_exporter": "python",
       "pygments_lexer": "ipython3",
       "version": "3.8.3"
      }
    },
    "nbformat": 4,
    "nbformat_minor": 4
    }
```