Deployment of Models on Cloud using Heroku

LECTURE 03

Deployment of Model (Titanic Passenger Survivial Predction System) on Cloud using Heroku

Authors:

Mr. Muhammad Ibtesam Arshad

Dr. Rao Muhammad Adeel Nawab

بِسۡمِ اللّٰہِ الرَّحۡمٰنِ الرَّحِیۡمِ

Lecture Outline

- Best Teaching and Learning Methodology of the World
- Using a Template-based Approach to Systematically Perform a Real-world Task
- Lecture Aim
- Titanic Passenger Survival Prediction Problem
- Steps - Deploying a Model (Titanic Passenger Survival Prediction System) on Cloud using Heroku
    - Obtain the Trained Model to be Deployed
    - Data and Code Organization
    - Develop Front-End Interface
    - Make Application Routing Handler File
    - Make Procfile
    - Make requirements.txt file
    - Uploading code to GitHub
    - Deployment of Model on Cloud using Heroku
    - Verify Deployment of Model on Heroku
- Stop Complaining! Stop Criticizing! Let's Start Contributing
- Lecture Summary

Titanic Passenger Survival Prediction Problem

SLIDE

Royal Mail Ship (RMS) Titanic – Brief Overview

- **RMS Titanic was a British passenger liner operated by the White Star Line that sank in the North Atlantic Ocean in the early morning hours of 15 April 1912, after striking an iceberg during her maiden voyage from Southampton to New York City**
- **Of the estimated 2,224 passengers and crew aboard, more than 1,500 died, making the sinking one of modern history's deadliest peacetime commercial marine disasters**
- **RMS Titanic was the largest ship afloat at the time she entered service[1]**

SLIDE

RMS Titanic – Main Features

- **Name of Ship**
  - **Royal Mail Ship (RMS) Titanic**
- **Manufactured**
  - **May 31, 1911**
- **Size**
  - **Length = 882 feet 9 inches (269.06 m)**
  - **Width = 92.5 feet (28.2 m)**
  - **Height = 104 feet (32 m)**
- **First Journey**
  - **April 10, 1912**
- **Source and Destination**
  - **Southampton, England to New York, USA**
- **Total Passengers On Board**
  - **2,208**
- **Passengers Survived**
  - **705**
- **Passengers Died**
  - **1,503**

[1] https://en.wikipedia.org/wiki/Titanic Last visited: 07-09-2020

SLIDE
RMS Titanic

Figure 01: Royal Mail Ship (RMS) Titanic [Source]

SLIDE
Lecture Focus

- **The main focus of this Lecture is developing a**
  - **Predictive System which can automatically predict whether a Passenger in RMS Titanic Survived or Not?**

SLIDE
Titanic Passenger Survival Prediction System

- **Real-world World**
  - **Titanic Passenger Survival**
- **Treated as**
  - **Supervised Machine Learning Problem**
- **Note**
  - **Titanic Passenger Survival Prediction Problem is treated as a**
    - **Binary Classification Problem because the**
      - **The main aim is to distinguish between Two Classes**
        - **Class 01 = Survived**
        - **Class 02 = Not Survived**
- **Goal**
  - **Learn an Input-Output Function**

- i.e. Learn from Input to predict the Output

SLIDE

Titanic Passenger Survival Prediction System – Task

- Given
  - A Passenger (Represented as Set of Attributes)
  - A pre-defined Set of Labels (Survived and Not Survived)
- Task
  - Automatically predict whether the Passenger Survived or Not

SLIDE

Titanic Passenger Survival Prediction System – Input and Output

- Input
  - A Passenger
- Output
  - Survived / Not Survived

SLIDE

Note

- In Kaggle Titanic Dataset, a Passenger is represented with many Attributes
- Kaggle Titanic Dataset
  - URL: https://www.kaggle.com/c/titanic
- For simplicity and to explain things more clearly
  - In this Lecture, we have represented a Passenger with Four Attributes

SLIDE

Titanic Passenger Survival Prediction System – Input Attributes

- In this Lecture, a Passenger is represented with the following Four Attributes
- Attribute 01 – PClass
  - Possible Value 01 = First
  - Possible Value 02 = Second
  - Possible Value 03 = Third
- Attribute 02 – Gender
  - Possible Value 01 = Female
  - Possible Value 02 = Male
- Attribute 03 – Sibling
  - Possible Value 01 = Zero
  - Possible Value 02 = One
  - Possible Value 03 = Two

    - Possible Value 04 = Three
- Attribute 04 – Embarked
    - Possible Value 01 = Cherbourg
    - Possible Value 02 = Southampton
    - Possible Value 03 = Queenstown

SLIDE

Titanic Passenger Survival Prediction System – Output Attributes

- In Titanic Dataset, there is One Output Attribute
    - Attribute 01 – Survived
        - Possible Value 01 = Yes
        - Possible Value 02 = No

SLIDE

Titanic Passenger Survival Prediction System – Summary (Input and Output)

- The following Table summarizes the Input and Output Attributes for Titanic Dataset

Attribute No.	Attribute Names	Possible Values	Data Types
1	PClass	First, Second, Third	Categorical
2	Gender	Male, Female	Categorical
3	Sibling	Zero, One, Two, Three	Categorical
4	Embarked	Southampton, Queenstown, Cherbourg	Categorical
5	Survived	Yes, No	Categorical

Table 01: Attributes of Dataset

SLIDE

Horrrrrrrraaaaaayyyyyyyyyyy! 🚩

- Alhamdulillah, we have understood the Titanic Passenger Survival Prediction Problem in detail
- In Sha Allah, in the next section, I will try to present the
    - Steps - Deploying a Model (Titanic Passenger Survival Prediction System) on Cloud using Heroku
- Note

    - **Always celebrate your achievements**
- **Remember**
    - **There are no such things as**
        - **Big Achievement**
        - **Small Achievement**
        - **Achievement is Achievement**

Steps - Deploying a Model (Titanic Passenger Survival Prediction System) on Cloud using Heroku

SLIDE

Steps - Deploying a Model (Titanic Passenger Survival Prediction System) on Cloud using Heroku

- **In sha Allah, I will follow the following Steps to deploy Model (Titanic Passenger Survival Prediction System) on Cloud using Heroku**
  - **Obtain the Trained Model to be Deployed**
  - **Data and Code Organization**
  - **Develop Front-End Interface**
  - **Make Application Routing Handler File**
  - **Make Procfile**
  - **Make requirements.txt file**
  - **Uploading code to GitHub**
  - **Deployment of Model on Cloud using Heroku**
  - **Verify Deployment of Model on Cloud using Heroku**

Obtain the Trained Model to be Deployed

SLIDE

Obtain the Trained Model to be Deployed

- In sha Allah, In this Lecture, I will use the following Trained Model (Titanic Passenger Survival Prediction System)
  - svc_trained_model.pkl
    - See svc_trained_model.pkl in Supporting Material
- The svc_trained_model.pkl was created in the following Lecture
  - Lecture 02 - Developing a Titanic Passenger Survival Prediction System using Train-Test Split Approach
  - Lecture Download Link
    - https://ilmoirfan.com/titanic-project/

SLIDE

Accuracy – Trained Model

- The Trained Model (svc_trained_model.pkl) obtained an
  - Accuracy Score of 80% on Test Data comprising of 20 instances
  - For details, See Lecture 02 - Developing a Titanic Passenger Survival Prediction System using Train-Test Split Approach
    - Lecture Download Link
      - https://ilmoirfan.com/titanic-project/

SLIDE

Input to Trained Model

- The Trained Model (svc_trained_model.pkl) will take the following Inputs and return the Prediction to the User on Unseen Real-time Data
- Input
  - Attribute 01 – PClass
    - Possible Value 01 = First
    - Possible Value 02 = Second
    - Possible Value 03 = Third
  - Attribute 02 – Gender
    - Possible Value 01 = Female
    - Possible Value 02 = Male
  - Attribute 03 – Sibling
    - Possible Value 01 = Zero
    - Possible Value 02 = One
    - Possible Value 03 = Two

        - **Possible Value 04 = Three**
    - **Attribute 04 – Embarked**
        - **Possible Value 01 = Cherbourg**
        - **Possible Value 02 = Southampton**
        - **Possible Value 03 = Queenstown**
- **Prediction**
    - **Attribute 01 – Survived**
        - **Possible Value 01 = Yes (Survived)**
          Possible Value 02 = No (Not Survived)

Data and Code Organization

SLIDE

Data and Code Organization

- The TODO: VERIFY Model Deployment Folder will contain the following Sub-Folders
  - Templates
    - This Sub-Folder will contain the Front-End Interface (Graphical User Interface) to Deploy the Model (Titanic Passenger Survival Prediction System) on Cloud using Heroku
  - Static
    - This Sub-Folder will contain the following Sub-Folders (which contains Files related to Front-End Interface (Graphical User Interface))
      - Images
      - CSS
      - JS

SLIDE

Flow Chart - Data and Organization

- The following Figure shows the Flow Chart for Data and Organization of Titanic Passenger Survival Prediction System deployment on Cloud using Heroku

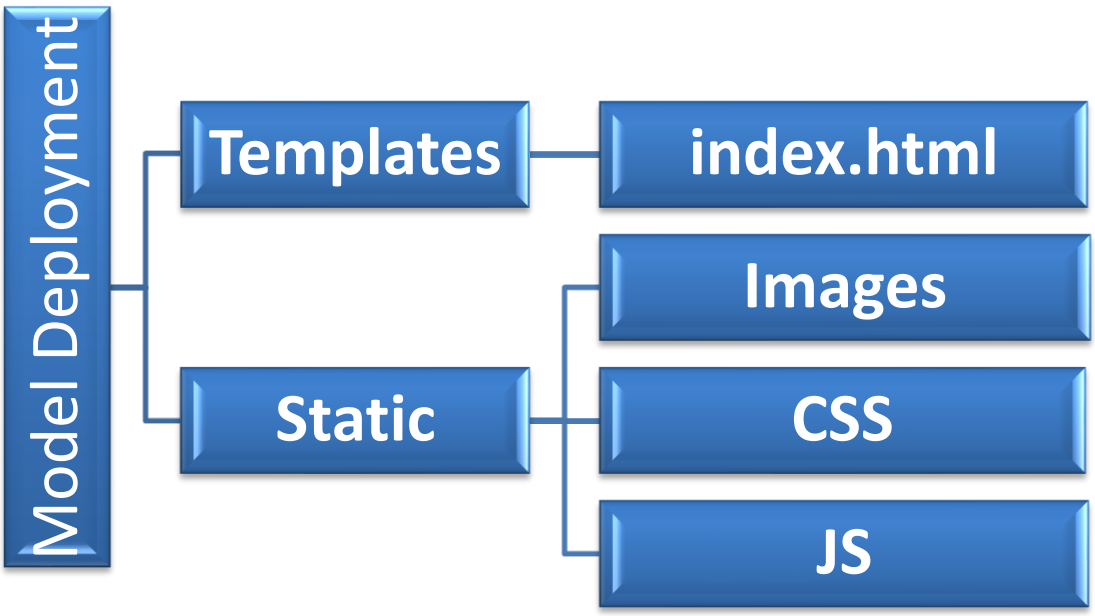

Figure 1 | Data and Code Organization

SLIDE

Steps - Data and Code Organization

- **In sha Allah, I will Organize my Data and Code in the following Steps**
- **Step 1: Create a new Folder in your Drive**
    - **In my case, New Folder is created on OS (C:) Drive**

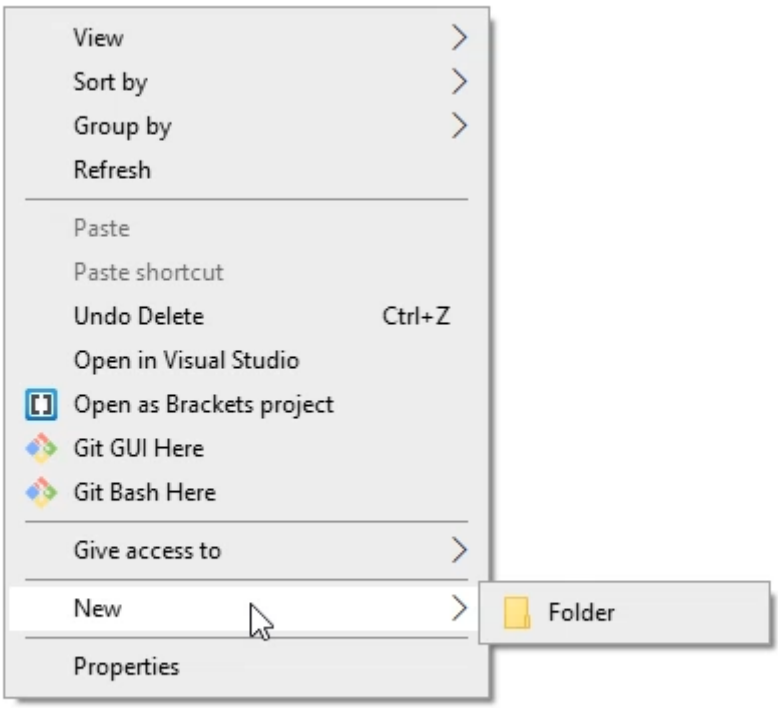

Figure 2 | Data and Code Organization

SLIDE

Steps - Data and Code Organization Cont…

- **Step 2: Rename your Folder**
    - **In my case, I renamed my Folder as**
        - **Model Deployment**

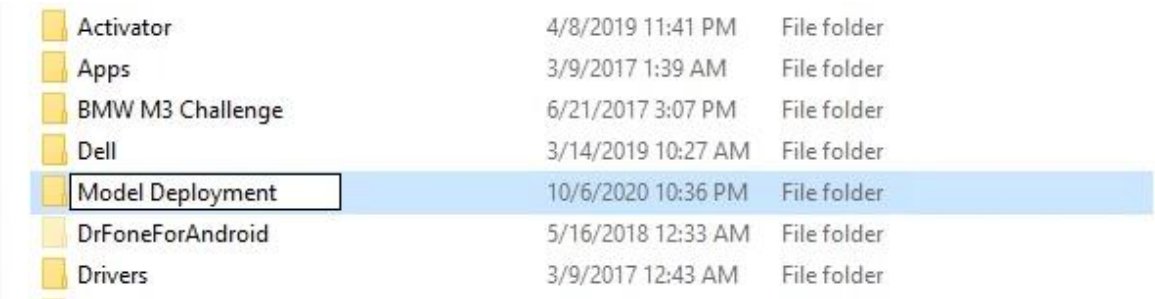

Figure 3 | Data and Code Organization

SLIDE

Steps - Data and Code Organization Cont…

- **Step 3: In Folder (Model Deployment)**
  - **Copy-Paste the svc_trained_model.pkl File**

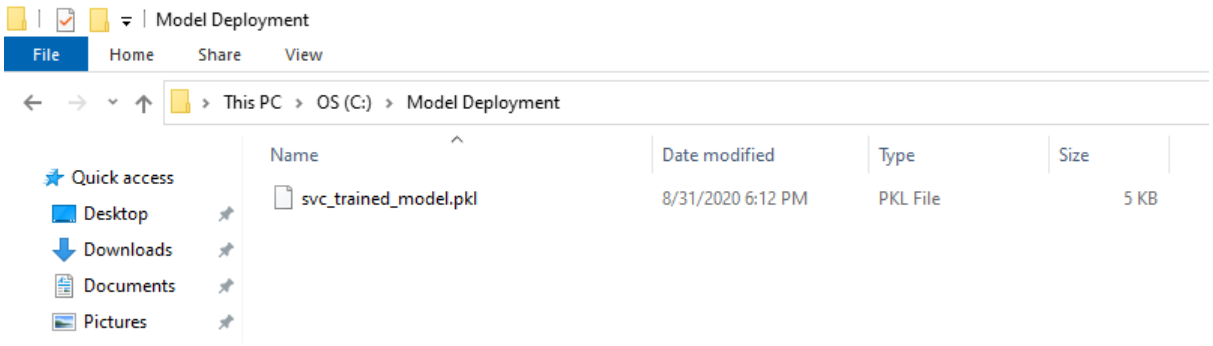

Figure 4 | Data and Code Organization

SLIDE

Steps - Data and Code Organization Cont…

- **Step 4: In Folder (Model Deployment), create the following Sub-Folders**
  - **Templates**
  - **Static**

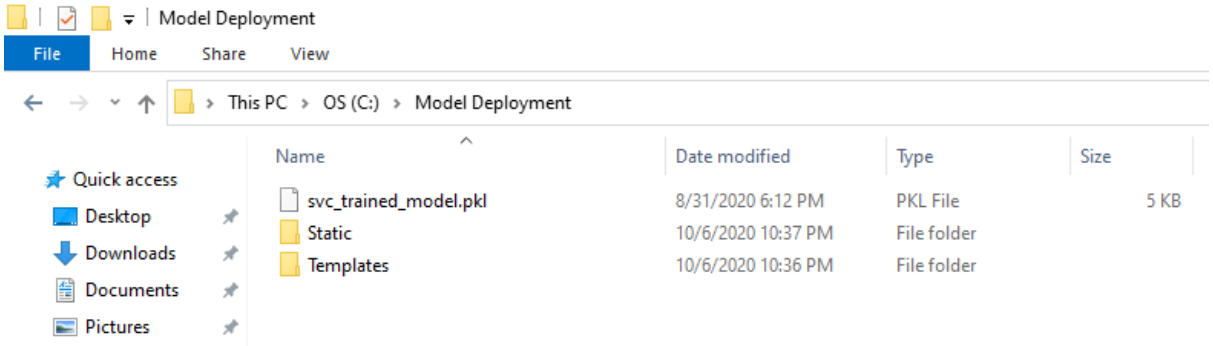

Figure 5 | Data and Code Organization

SLIDE

Steps - Data and Code Organization Cont…

- **In Static Sub-folder, create the following Sub-sub-folders**
  - **Images**
  - **CSS**
  - **JSS**

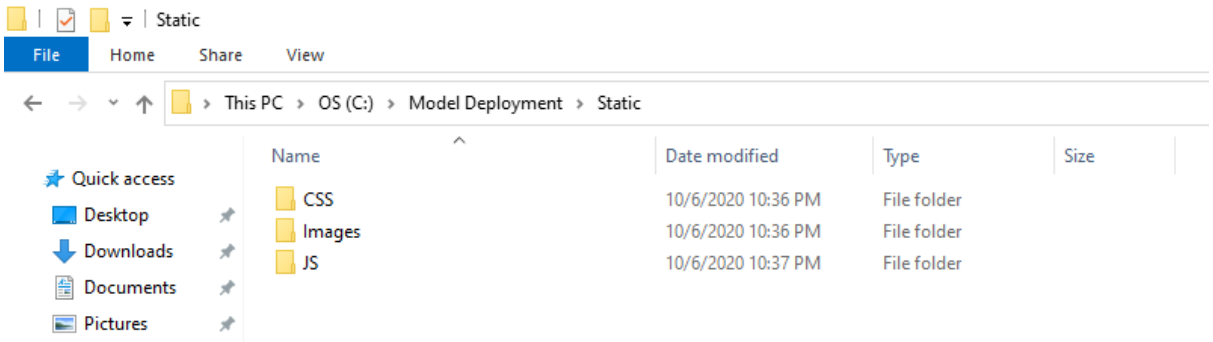

Figure 6 | Data and Code Organization

SLIDE

Horrrrrrrraaaaaaayyyyyyyyyyy! 🚩

- **Alhumdulilah, Data and Code Organization is completed successfully**
- **In sha Allah, in the next Section, I will try to explain how to Develop Front-End Interface (Graphical User Interface) for Deployment of Titanic Passenger Survival Prediction System**
- **Note**
  - **Always celebrate your achievements**
- **Remember**
  - **There are no such things as**
    - **Big Achievement**
    - **Small Achievement**
  - **Achievement is Achievement** 😊

Developing a Front-End Interface for Deployment of Titanic Passenger Survival Prediction System

System Settings	
Developer Name	Mr. Muhammad Ibtesam Arshad
Programming Languages	HTML5, CSS 3, JS ES9, Python 3.7.0b2
Software	Jupyter Notebook 6.0.1, Visual Studio Code 1.49.2
Browser	Google Chrome 85.0.4183.102
Date	02-Sept-2020

SLIDE
Development of Front-End Interface

- Insha Allah, I will develop a Front-End Interface for Deployment of Titanic Passenger Survival Prediction System (using Visual Studio Code 1.49.2) in two main Steps
    1. Create an index.html File
    2. Write Fond-End Interface Code in index.html File

SLDIE
TIP – Popular IDEs for Development of Front-End Interfaces

- Some of the popular and widely used IDEs for developing Front-End Interface are as follows
    - Visual Studio Code 1.49.2
    - Brackets 1.14.2
    - Notepad++ 7.8.7

Create an index.html File

SLIDE
Create an index.html File

- **Insha Allah, I will create an index.html File in the following Steps**

SLIDE
Steps - Create an index.html File

- **Step 1: Open Visual Studio Code**
  - **Step 1.1: Click on Open Folder**
    - **A New Wizard will be Displayed**

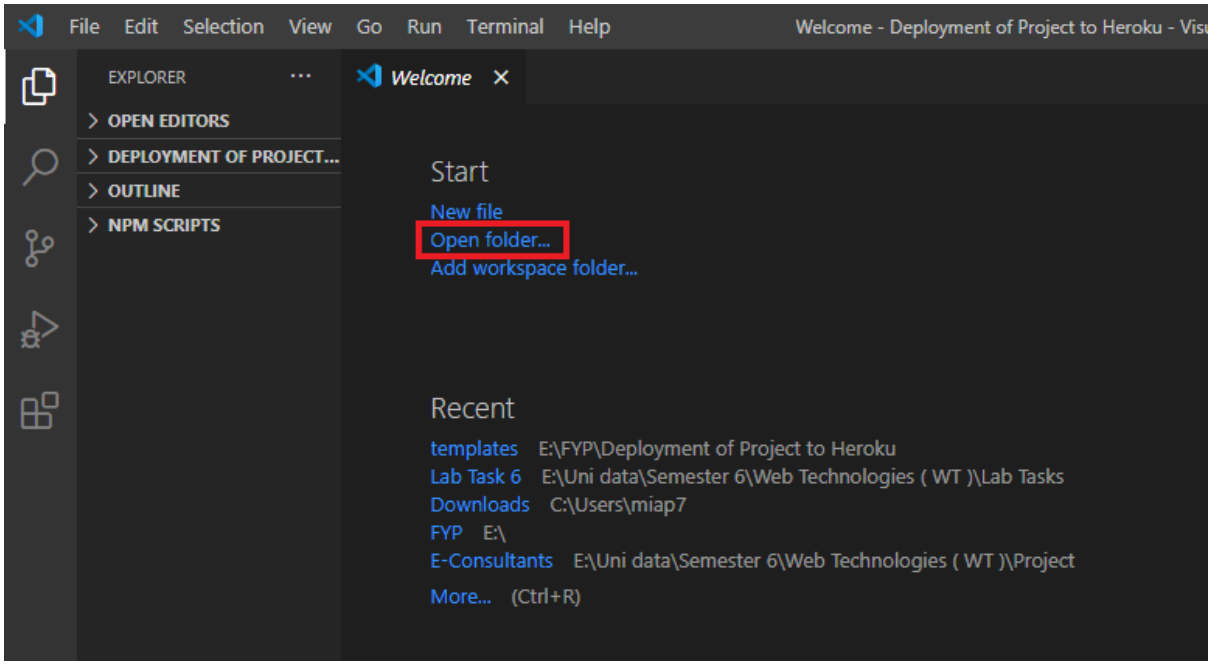

Figure 1 | Development of Front-End Interface

SLIDE
Steps - Create an index.html File Cont…

- **Step 2: Open Templates Folder to Load Your Project in Visual Studio Code**
  - **In my case, Path to Templates Folder is as follows**
    - **C:/Model Deployment/Templates**

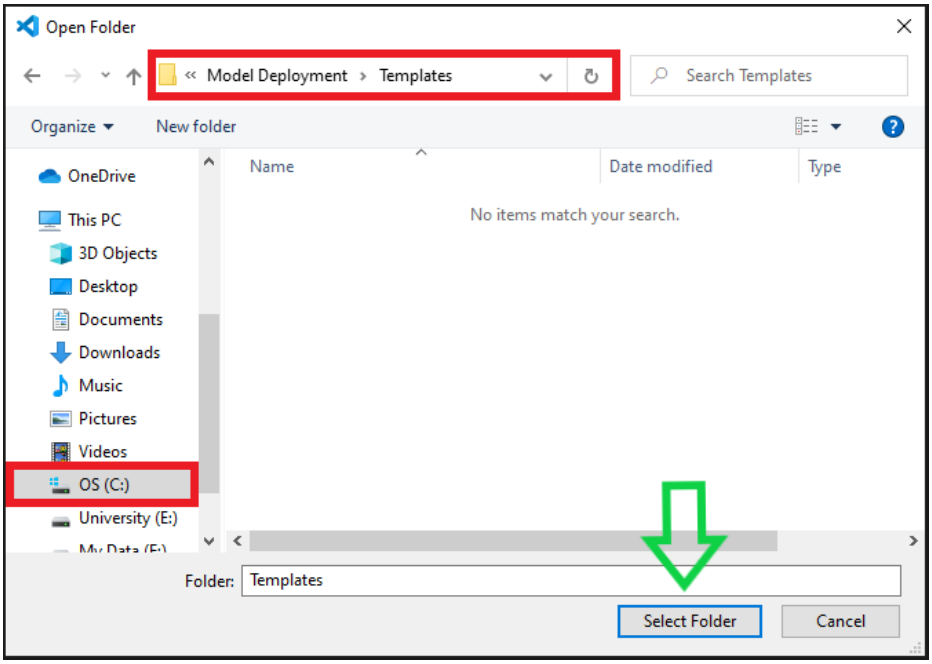

Figure 2 | Development of Front-End Interface

SLIDE

Steps - Create an index.html File Cont…

Step 3: Create index.html File

Step 3.1: Click on  **to create a New File**

Step 3.2: Rename that file to index.html

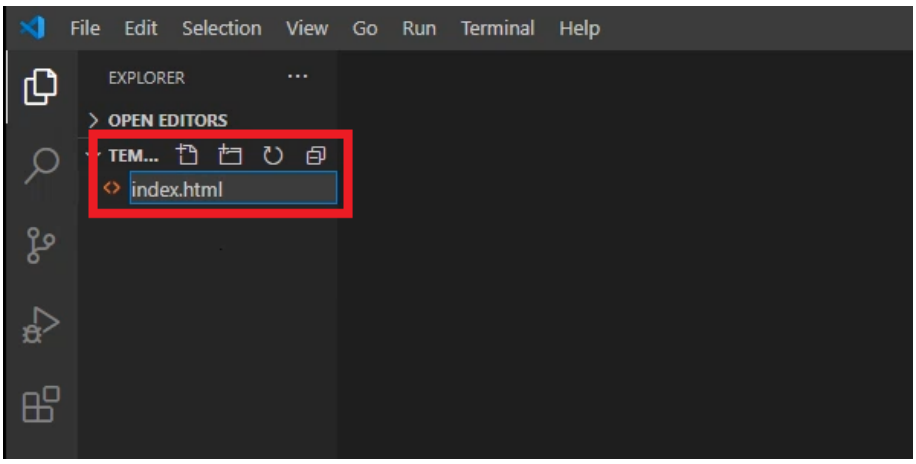

Figure 3 | Development of Front-End Interface

SLIDE

Steps - Create an index.html File Cont…

- **Alhumdulilah, index.html File has been successfully created**
- **In sha Allah, in the next Section, I will write Code in index.html using HTML, CSS, and Java Script (JS) to develop my Font-End Interface for Deployment of Model (Titanic Passenger Survival Prediction System) on Cloud using Heroku**

Web Development Basics for Front End Interface

SLIDE

Web Development Basics for Front End Interface

- **Web Development – Definition**
  - **Web development is the building and maintenance of websites; It's the work that happens behind the scenes to make a website look great, work fast and perform well with a seamless user experience. Web developers, or 'devs', do this by using a variety of coding languages.**
- **Frameworks**
  - **Following are some of the popular commonly used frameworks**
    - **Ruby on Rails**
    - **ExpressJS**
    - **Laravel**
    - **React**
    - **Bootstrap**

Write Fond-End Interface Code in index.html File

SLIDE
Write Fond-End Interface Code in index.html File

- **In sha Allah, my Front-End Interface will comprise of the following six Parts**
  - **Part 01: Making Basic Structure and Adding Title to the Project**
  - **Part 02: Create a Container and Add Two Cards (Sections) into Container**
    - **Part 2.1: Card 01 (Left Card): Project Overview and Instructions to Run the Deployed ML Model**
    - **Part 2.2: Card 02 (Right Card): Prediction System**
  - **Part 03: Populate the Left Card**
    - **Part 3.1: Add Project Overview Information**
    - **Part 3.2: Add Instructions to Run the Deployed ML Model**
  - **Part 04: Populate the Right Card**
    - **Part 4.1: Add Drop Down Menus to Take Input from User**
    - **Part 4.2: Add Predict Button (to Generate Prediction based on Input Given in Part 4.1)**
  - **Part 05: Create Modals for Instructions and Result Display**
  - **Part 06: Setting Events on Clicks and Actions**

Part 01: Making Basic Structure and Adding Title to the Project

---

SLIDE

Part 01: Making Basic Structure and Adding Title to the Project

- **In index.html you need to create Front-End Interface using HTML, CSS, and Java Script (JS)**
- **In sha Allah, I will make the Basic Structure and add Title to the Project in index.html File in the following Steps**

SLIDE

Steps - Making Basic Structure and Adding Title to the Project

- **Step 01: Create Basic Structure for HTML File as shown below**

HTML Code – Basic Structure

```
<!DOCTYPE html>
<html>

   <head>

   </head>

   <body>

      <script>

      </script>

   </body>

</html>
```

Figure 1 | Making Basic Structure and Adding Title to the Project

SLIDE

Steps - Making Basic Structure and Adding Title to the Project Cont…

- **Step 02: In <head> Tag**
  - **Create <title> Tag (to create Title of the Project)**
- **In my case, Title of the Project is Titanic Passenger Survival Prediction System**

HTML Code - Title to Project

```
<title>Titanic Passenger Survival Prediction System</title>
```

Figure 2 | Making Basic Structure and Adding Title to the Project

Output

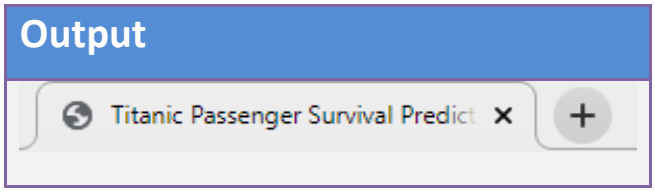

Figure 2 | Output

SLIDE

Steps - Making Basic Structure and Adding Title to the Project Cont…

- **Step 03: In <head> Tag**
  - **Import bootstrap and other Supporting Material using CDN**
- **In my case, imports are as follows**

HTML Code - Imports

```
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1" />
    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.3
.1/css/bootstrap.min.css" />
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/4.7.0/css/font-awesome.min.css" />

    <!-- Custom Stylesheet-->
    <link rel="stylesheet" href="{{ url_for('Static',filename='styles/mystyle.
css') }}" />


    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min
.js"></script>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/p
opper.min.js"></script>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/prefixfree/1.0.7/prefi
xfree.min.js"></script>
    <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.
min.js"></script>
    <script src="http://use.fontawsome.com/releases/v5.0.8/js/all.js"></script
>

    <!-- Custom Scripting-->
    <script src="{{ url_for('Static',filename='js/myscript.js') }}"></script>
```

Figure 3 | Making Basic Structure and Adding Title to the Project

SLIDE

Steps - Making Basic Structure and Adding Title to the Project Cont…

- **Step 04: Download Background Image from the following Link and save it in Images Folder (which is a Sub-folder of Static Folder)**
  - **https://wallpaper-house.com/group/titanic-backgrounds/index.php**
- **In my case, name of Background Image is as follows**
  - **titanicbackground.jpg**

Figure 4 | Making Basic Structure and Adding Title to the Project

SLIDE

Steps - Making Basic Structure and Adding Title to the Project Cont…

- **Step 05: In <body> Tag**
  - **Add styling for Background Image (titanicbackground.jpg) to**
    - **Insert titanicbackground.jpg and**
    - **Set size of titanicbackground.jpg**

HTML Code – Setting Background Image

```
<body style="background:url(../Static/Images/titanicbackground.jpg);background
-size: cover;">
```

Figure 5 | Making Basic Structure and Adding Title to the Project

Output

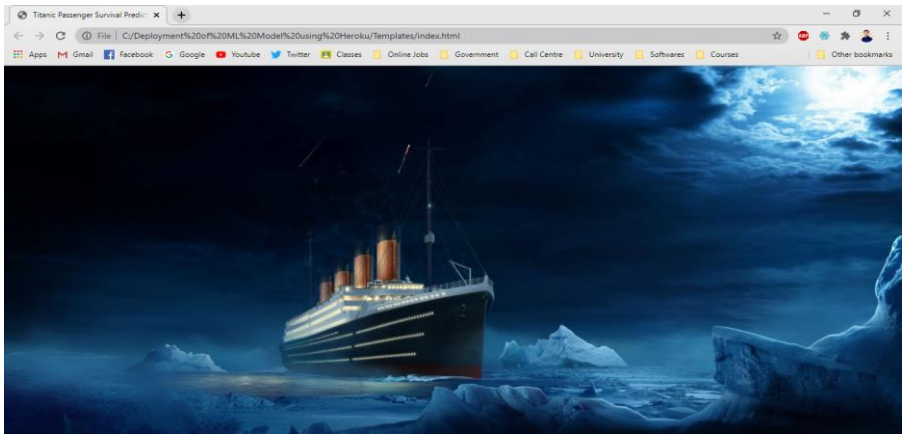

Figure 5 | Output

Part 02: Create a Container and Add Two Cards (Sections) into Container

---

SLIDE

Part 02: Create a Container and Add Two Cards (Sections) into Container

- **In index.html you need to create Container and divide that container into following two sections**
  - **Left Card (Section) - Project Overview and Instructions to Run the Deployed ML Model**
  - **Right Card (Section) - Prediction System**

- **In sha Allah, I will create a Container using Bootstrap class and divide that container into above two sections**

SLIDE

Steps - Create a Container and Add Two Cards (Sections) into Container

- **Step 01: In <body> Tag**
  - **Add a div and give container class to the div and to give margin to the container from top and bottom we will add classes of mt-5 and mb-5 respectively to give 50px (suitable) margin from top and bottom**
  - **Add another div and give class row to add row in that container**

HTML Code – Creating Container and Adding Row to it

```
<div class="container mt-5 mb-5">

    <div class="row">

    </div>

</div>
```

Figure 1 | Create a Container and Add Two Cards (Sections) into Container

SLIDE

Steps - Create a Container and Add Two Cards (Sections) into Container

Cont…

- **Step 01: In container (div) create further two divs within row (div) and give class card to those created divs to divide container into two sections**

HTML Code – Create Two Cards (Sections)

```
<!-- Main div -->
<div class="container mt-5 mb-5">
    <div class="row">

        <!-- Left Card for Project Introduction -->
        <div class="card col-lg-7 text-
center" style="color: white; background: rgba(0, 0, 0, 0.7);">

        </div>

        <!-- Right Card for System Inputs -->
        <div class="col-lg-5 text-
center" style="background: rgba(247, 223, 227, 0.7)">

        </div>

    </div>
</div>
```

Figure 2 | Create a Container and Add Two Cards (Sections) into Container

---
Part 03: Populate the Left Card

SLIDE
Part 03: Populate the Left Card

- **In Left Card we will add following**
  - **Add Project Overview Information**
  - **Add Instructions to Run the Deployed ML Model**
- **In sha Allah, I will add Project Overview Information and Instructions to Run the Deployed ML Model**

SLIDE
Steps - Populate the Left Card

- **Step 01: In Left Card Tag we will add Title Heading, Salutation, Instructions List and Read Instructions button which will display the detailed instructions and steps to use the system**

HTML Code – Polpulating Left Card

```
<!-- Title Heading -->
<h1 class="m-5">
  <strong>Titanic Passenger Survival Prediction System</strong>
</h1>

<!-- Salutaion -->
<p class="ml-5" style="font-weight: bolder; color: green; text-align: left">
                    Asslam-o-Alaikum !</p>

<!-- Instructions -->
<ul class="ml-4 mr-4" style="text-align: justify">

  <li>Titanic was a british passenger liner that sank in the North
Atlantic Ocean in the early morning hours of 15 April 1912 after
striking an iceberg</li>

  <li>Of the estimated 2,224 passengers and crew aboard, more than 1,500
died</li>

  <li>The main aim of this project is to predict whether a passenger
aboard on Titanic Ship survived or not?</li>
</ul>

<!-- Read Instructions button -->
<button class="btn btn-secondary mt-4" style="width: 200px; margin-
left: 50px; margin-bottom: 50px" data-toggle="modal" data-
target="#instructionmodal">Read Instructions
</button>
```

Figure 1 | Populate the Left Card

Output

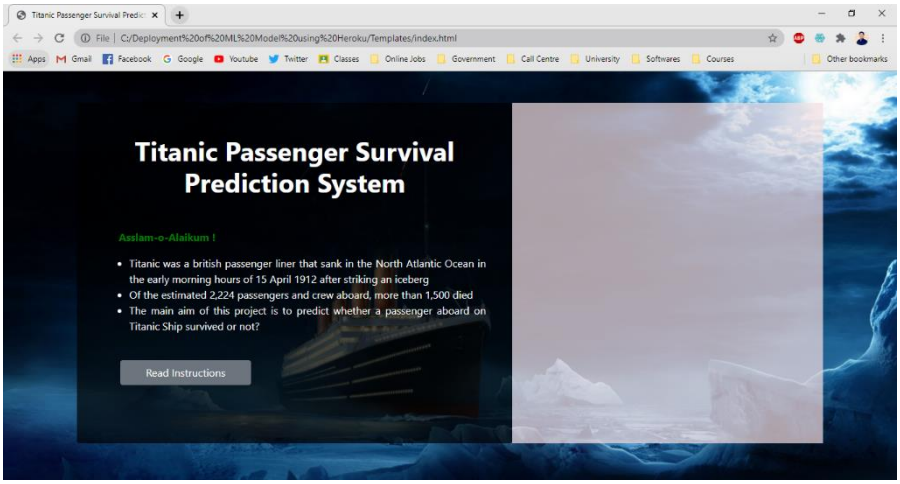

Figure 1 | Output

Part 04: Populate the Right Card

SLIDE

Part 04: Populate the Right Card

- **In Right Card we will add following**
  - **Add Drop Down Menus to Take Input from User**
  - **Add Predict Button (to Generate Prediction based on Input Given)**
- **In sha Allah, I will add Drop Down Menus to Take Input from User and a Predict Button to Generate Prediction based on Input Given**

SLIDE

Steps - Populate the Left Card

- **Step 01: In Left Card Tag we will add a Heading, four Drop Down Menus (PClass, Gender, Siblings, Embarked) and a Predict Button to display the Prediction Result on the basis of user inputs**

HTML Code – Polpulating Right Card

```
<div>
  <h2 class="m-5" style="color: black"><strong>Predict Survival</strong></h2>
</div>

<div id="inputs-form">
  <form action="/" method="POST">

      <!-- PClass Dropdown -->
      <div class="form-group row">
         <label class="col-lg-4 col-4 offset-lg-1 col-form-label"
         style="font-weight: bolder; font-size: large">PClass</label>
         <div class="col-lg-6 col-xl-5 col-6">
            <div class="dropdown">
               <select class="form-control btn-
secondary" name="pclass" id="pclass" style="background-
color: rgb(0, 0, 0, 0.5);border-width: 3px;border-color: #f0526c;color: white;
height: 40px;" onchange="checkSelection(this)" required>
                  <option selected disabled value="">Select PClass</option>
                  <option value="1">First Class</option>
                  <option value="2">Second Class</option>
                  <option value="3">Third Class</option>
               </select>
            </div>
         </div>
      </div>

      <!-- Gender Dropdown -->
      <div class="form-group row">
```

```
        <label class="col-lg-4 col-4 offset-lg-1 col-form-label"
        style="font-weight: bolder; font-size: large">Gender</label>
        <div class="col-lg-6 col-xl-5 col-6">
           <div class="dropdown">
              <select class="form-control btn-
secondary" name="gender" id="gender" style="background: rgb(0, 0, 0, 0.5);
border-width: 3px;border-color: #f0526c;color: white;height: 40px;
" onchange="checkSelection(this)" required>
                 <option selected disabled value="">Select Gender</option>
                 <option value="1">Male</option>
                 <option value="0">Female</option>
              </select>
            </div>
        </div>
     </div>

     <!-- Siblings Dropdown -->
     <div class="form-group row">
        <label class="col-lg-4 col-4 offset-lg-1 col-form-label"
        style="font-weight: bolder; font-size: large">Siblings</label>
        <div class="col-lg-6 col-xl-5 col-6">
           <div class="dropdown">
              <select class="form-control btn-
secondary" name="siblings" id="siblings" style="background: rgb(0, 0, 0, 0.5);
border-width: 3px;border-color: #f0526c;color: white;height: 40px;
" onchange="checkSelection(this)" required>
                 <option selected disabled value="">Select Siblings</option>
                 <option value="0">None</option>
                 <option value="1">One</option>
                 <option value="2">Two</option>
                 <option value="3">Three</option>
              </select>
           </div>
        </div>
     </div>

     <!-- Embarked Dropdown -->
     <div class="form-group row">
        <label class="col-lg-4 col-4 offset-lg-1 col-form-label"
        style="font-weight: bolder; font-size: large">Embarked</label>
        <div class="col-lg-6 col-xl-5 col-6">
           <div class="dropdown">
              <select class="form-control btn-
secondary" name="embarked" id="embarked" style="background: rgb(0, 0, 0, 0.5);
border-width: 3px;border-color: #f0526c;color: white;height: 40px;
" onchange="checkSelection(this)" required>
                 <option selected disabled value="">Select Embarked</option>
                 <option value="0">Cherbourg</option>
                 <option value="1">Queenstown</option>
                 <option value="2">Southampton</option>
              </select>
           </div>
        </div>
     </div>

     <!-- Predict Button -->
     <div class="form-group row mt-5">
```

```
          <div class="col-lg-8 offset-lg-2 col-9 offset-1 mb-5">
             <button type="submit" class="btn btn-
success" id="calculateButton" style="width: 100%">Predict
             </button>
          </div>
       </div>
    </form>
</div>
```

Figure 1 | Populate the Left Card

Output

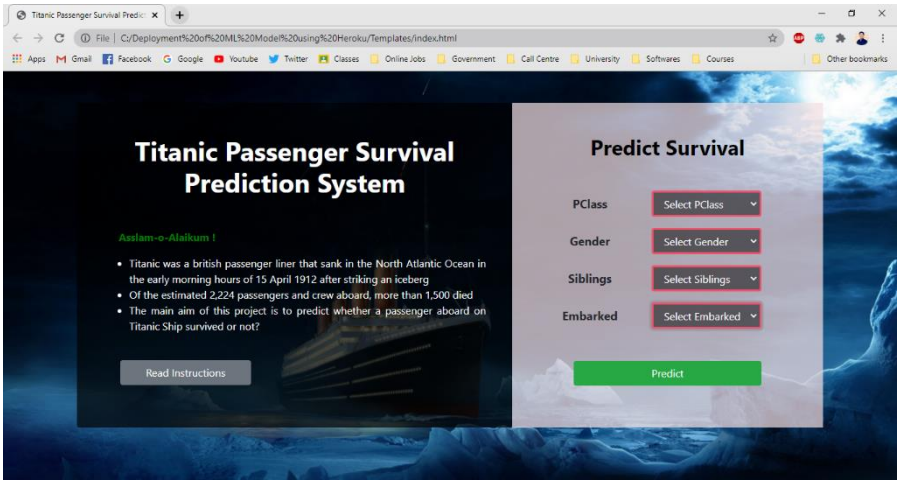

Figure 1 | Output

Part 05: Create Modals for Instructions and Result Display

SLIDE

Part 05: Create Modals for Instructions and Result Display

- **You need to create following two modals to popup the information to the user**
  - **Instruction Modal – This modal will display the instructions to the user about the usage of system**
  - **Result Modal – This modal will display the predicted result to the user**

- **In sha Allah, I will create two modals to display the instructions and predicted output to the user**

SLIDE

Steps - Create Modal for Instructions

- **Step 01: Create a basic modal for Instructrions Details as given below**

HTML Code – Create Modal for Instructions

```
<!-- Instruction Modal -->
<div class="modal fade" id="instructionmodal" tabindex="-
1" role="dialog" aria-labelledby="modelTitleId" aria-hidden="true">
   <div class="modal-dialog" role="document">
      <div class="modal-content">
         <div class="modal-header">

         </div>

         <div class="modal-body">

         </div>

      </div>
   </div>
</div>
```

Figure 1 (a) | Create Modal for Instructions

- **Step 02: In Instruction Modal header insert the code below to display the heading and an icon to hide the modal**

HTML Code – Create Modal for Instructions

```
<h4 class="modal-title" style="margin-left: 35%">
    <strong>Instructions</strong>
</h4>

<button type="button" class="close" data-
dismiss="modal" aria-label="Close">
    <span aria-hidden="true">&times;</span>
</button>
```

Figure 1 (b) | Create Modal for Instructions

- **Step 03: In Instruction Modal body insert the code below to display the collapsable inputs with their details**

HTML Code – Create Modal for Instructions

```
<!-- Instruction Heading -->
<div class="w-100 text-
center" style="background: #009cff; color: white; height: 50px">
    <h5 style="padding-top: 10px">Input Attributes Details</h5>
</div>

<!-- Collapsable System Input Attributes in Instruction Modal -->
<div id="accordion">
    <div class="card mt-2" style="background: rgb(173, 170, 170)">
        <div class="card-
header" id="pclassheading" style="display: inline-flex;height: 40px;align-
content: center;cursor: pointer;" data-toggle="collapse" data-
target="#pclasscollapse" aria-expanded="false"
            aria-controls="collapseOne">
            <i class="fa fa-plus-circle">
                <b class="ml-3" style="font-family: sans-serif">PClass -
 Passenger Class</b>
            </i>
        </div>
        <div id="pclasscollapse" class="collapse" aria-
labelledby="pclassheading"
            style="background: white" data-parent="#accordion">
            <div class="card-body">
                Possible Values of PClass are :
                <ul>
                    <li>First Class</li>
                    <li>Second Class</li>
                    <li>Third Class</li>
                </ul>
```

```
            </div>
        </div>
    </div>
    <div class="card mt-1" style="background: rgb(173, 170, 170)">
        <div class="card-
header" id="genderheading" style="display: inline-flex;height: 40px;align-
content: center;cursor: pointer;" data-toggle="collapse" data-
target="#gendercollapse" aria-expanded="false"
            aria-controls="collapseOne">
            <i class="fa fa-plus-circle">
                <b class="ml-3" style="font-family: sans-serif">Gender</b>
            </i>
        </div>
        <div id="gendercollapse" class="collapse" aria-
labelledby="genderheading"
            style="background: white" data-parent="#accordion">
            <div class="card-body">
                Possible Values of Gender are :
                <ul>
                    <li>Male</li>
                    <li>Female</li>
                </ul>
            </div>
        </div>
    </div>
    <div class="card mt-1" style="background: rgb(173, 170, 170)">
        <div class="card-
header" id="siblingsheading" style="display: inline-
flex;height: 40px;align-content: center;cursor: pointer;" data-
toggle="collapse" data-target="#siblingscollapse" aria-expanded="false"
            aria-controls="collapseOne">
            <i class="fa fa-plus-circle">
                <b class="ml-3" style="font-family: sans-
serif">Siblings</b>
            </i>
        </div>
        <div id="siblingscollapse" class="collapse" aria-
labelledby="siblingsheading"
            style="background: white" data-parent="#accordion">
            <div class="card-body">
                Possible Values of Siblings are :
                <ul>
                    <li>None</li>
                    <li>One</li>
                    <li>Two</li>
                    <li>Three</li>
                </ul>
            </div>
        </div>
    </div>
    <div class="card mt-1" style="background: rgb(173, 170, 170)">
        <div class="card-
header" id="embarkedheading" style="display: inline-
flex;height: 40px;align-content: center;cursor: pointer;" data-
toggle="collapse" data-target="#embarkedcollapse" aria-expanded="false"
            aria-controls="collapseOne">
            <i class="fa fa-plus-circle">
```

```
                <b class="ml-3" style="font-family: sans-
serif">Embarked</b>
            </i>
        </div>
        <div id="embarkedcollapse" class="collapse" aria-
labelledby="embarkedheading"
            style="background: white" data-parent="#accordion">
            <div class="card-body">
                Possible Values of Embarked are :
                <ul>
                    <li>Cherbourg</li>
                    <li>Southampton</li>
                    <li>Queenstown</li>
                </ul>
            </div>
        </div>
    </div>
</div>
```

Figure 1 (c) | Create Modal for Instructions

- **Step 04: In Instruction Modal body insert the code below to add Steps to follow for prediction**

HTML Code – Create Modal for Instructions

```
<div class="w-100 text-center mt-
3" style="background: #009cff; color: white; height: 50px">
    <h5 style="padding-top: 10px">Steps to Follow</h5>
</div>

<!-- Steps of Prediction -->
<div class="mt-3 ml-3">
    <span style="font-weight: bolder">Step 1 : </span> Select
    values of
    <span style="font-
weight: bolder; color: #009cff">PClass, Gender, Siblings, Embarked
    </span><br />
    <span style="font-weight: bolder">Step 2 : </span> Click on
    <span style="font-
weight: bolder; color: #009cff">Predict</span>
    Button<br />
    <span style="font-weight: bolder">Step 3 : </span> Our Titanic
    Passenger Survival Prediction System will predict whether
    passenger
    <span style="font-
weight: bolder; color: green">Survived</span>
    or <span style="font-weight: bolder; color: red">Not </span> ?
</div>
```

Figure 1 (d) | Create Modal for Instructions

Output

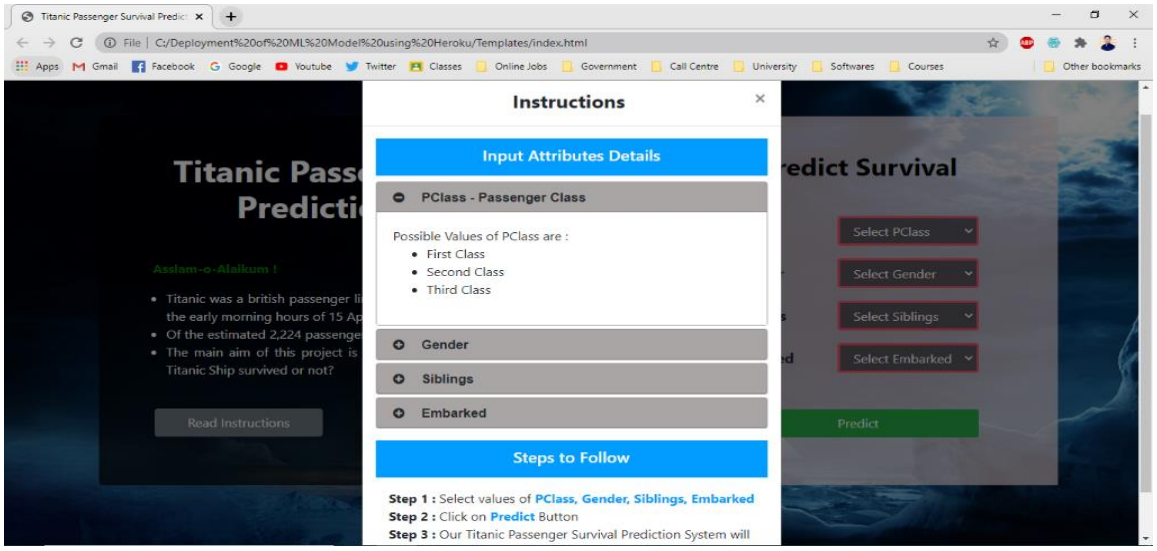

Figure 1 | Output

SLIDE

Create Modal for Result

- **Step 01: Create a basic modal for Result Display as given below**

HTML Code – Create Modal for Result

```
<!-- Result Modal -->
<div class="modal fade" id="resultmodal" tabindex="-
1" role="dialog" aria-labelledby="modelTitleId"
    aria-hidden="true">
    <div class="modal-dialog" role="document">
        <div class="modal-content">
            <div class="modal-header">

            </div>
            <div class="modal-body text-center">

            </div>
        </div>
    </div>
</div>
```

Figure 2 (a) | Create Modal for Result

- **Step 02: In Result Modal header insert the code below to display the heading and an icon to hide the modal**

HTML Code – Create Modal for Result

```
<h4 class="modal-title" style="margin-left: 26%; margin-
right: 17%">
    <strong>Prediction of Model</strong>
</h4>
<button type="button" class="close" data-dismiss="modal" aria-
label="Close">
    <span aria-hidden="true">&times;</span>
</button>
```

Figure 2 (b) | Create Modal for Result

- **Step 03: In Result Modal body insert the code below to add a button to control the display of provided inputs**

HTML Code – Create Modal for Result

```
<button class="btn btn-success" type="button" data-toggle="collapse"
    data-target=".multi-collapse" aria-expanded="false"
    aria-
controls="multiCollapseExample1 multiCollapseExample2" id="inputhandler">
    Click to view inputs
</button>
```

Figure 2 (c) | Create Modal for Result

- **Step 04: In Result Modal body insert the code below to display the inputs and an arrow pointing downwards**
  - **Note that we need to write following text in card body to display the forwarded inputs from application routing handler**
    - **{{pclass}} for PClass**
    - **{{gender}} for Gender**
    - **{{siblings}} for Siblings**
    - **{{embarked}} for Embarked**

HTML Code – Create Modal for Result

```
<!-- Display of Forwarded Inputs In Result Modal -->
<div class="row mt-3">
```

```
    <div class="col-6">
        <div class="collapse multi-collapse" id="multiCollapseExample1">
            <div class="card">
                <div class="card-header" style="background-
color: #009cff; color: white">
                    <strong>PClass</strong>
                </div>
                <div class="card-body">{{pclass}}</div>
            </div>
        </div>
    </div>
    <div class="col-6">
        <div class="collapse multi-collapse" id="multiCollapseExample2">
            <div class="card">
                <div class="card-header" style="background-
color: #009cff; color: white">
                    <strong>Gender</strong>
                </div>
                <div class="card-body">{{gender}}</div>
            </div>
        </div>
    </div>
    <div class="col-6 mt-2">
        <div class="collapse multi-collapse" id="multiCollapseExample3">
            <div class="card">
                <div class="card-header" style="background-
color: #009cff; color: white">
                    <strong>Siblings</strong>
                </div>
                <div class="card-body">{{siblings}}</div>
            </div>
        </div>
    </div>
    <div class="col-6 mt-2">
        <div class="collapse multi-collapse" id="multiCollapseExample4">
            <div class="card">
                <div class="card-header" style="background-
color: #009cff; color: white">
                    <strong>Embarked</strong>
                </div>
                <div class="card-body">{{embarked}}</div>
            </div>
        </div>
    </div>
</div>

<!-- Arrow pointing downwards -->
<div class="row mt-3 collapse multi-collapse">
    <i class="fa fa-arrow-down" style="padding-left: 43%; font-
size: 70px"></i>
</div>
```

Figure 2 (d) | Create Modal for Result

- **Step 05: In Result Modal body insert the following code to display the forwarded prediction result**
    - **Note that we need to insert {{predicted_result}} text to display the forwarded output from application routing handler**

HTML Code – Create Modal for Result

```
<!-- Display of predicted result -->
<div class="result-box mt-3">
    <p id="result">{{predicted_result}}</p>
</div>
```

Figure 2 (e) | Create Modal for Result

Output

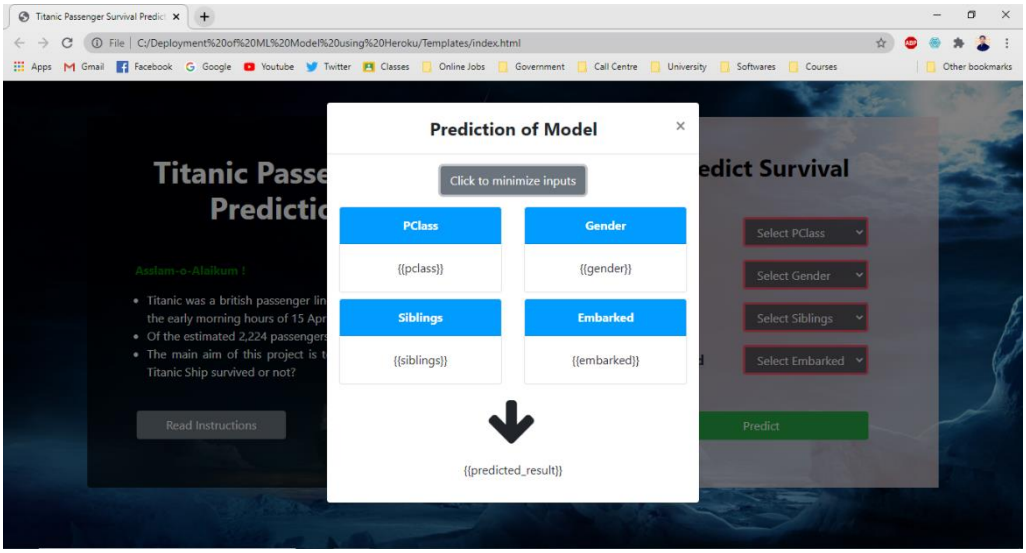

Figure 2 | Output

Part 06: Setting Events on Clicks and Actions

SLIDE

Steps - Setting Events on Clicks and Actions

- You need to create events on clicks and different actions
    - Handle Instruction Modal Collapse Icons – In this part we need to handle collapse icons while collapse expanding
    - Handle Result Modal – In this part we need to display the result modal only when the predicted result will be received in modal
    - Handle Input Control Button – In this part we have to control the content and style of Input Control Button
    - Handle Predicted Result – In this part we need to control the style of predicted result

- In sha Allah, I will create four events as described above so we need to add <script> tag along with its closing tag </script> to create events

SLIDE

Handle Instruction Modal Collapse Icons

- Step 01: We will handle collapse icons to switch between while collapse expanding

Javascript Code – Handle Instruction Modal Collapse Icons

```
// Handle PClass collapse on click
$("#pclassheading").click(function () {
    if ($("#pclassheading").attr("aria-expanded") == "false") {
        $("#pclassheading i").removeClass("fa-plus-circle");
        $("#pclassheading i").addClass("fa-minus-circle");
    }
    if ($("#pclassheading").attr("aria-expanded") == "true") {
        $("#pclassheading i").removeClass("fa-minus-circle");
        $("#pclassheading i").addClass("fa-plus-circle");
    }
});

// Handle Gender collapse on click
$("#genderheading").click(function () {
    if ($("#genderheading").attr("aria-expanded") == "false") {
        $("#genderheading i").removeClass("fa-plus-circle");
        $("#genderheading i").addClass("fa-minus-circle");
    }
    if ($("#genderheading").attr("aria-expanded") == "true") {
        $("#genderheading i").removeClass("fa-minus-circle");
```

```
        $("#genderheading i").addClass("fa-plus-circle");
    }
});

// Handle Siblings collapse on click
$("#siblingsheading").click(function () {
    if ($("#siblingsheading").attr("aria-expanded") == "false") {
        $("#siblingsheading i").removeClass("fa-plus-circle");
        $("#siblingsheading i").addClass("fa-minus-circle");
    }
    if ($("#siblingsheading").attr("aria-expanded") == "true") {
        $("#siblingsheading i").removeClass("fa-minus-circle");
        $("#siblingsheading i").addClass("fa-plus-circle");
    }
});

// Handle Embarked collapse on click
$("#embarkedheading").click(function () {
    if ($("#embarkedheading").attr("aria-expanded") == "false") {
        $("#embarkedheading i").removeClass("fa-plus-circle");
        $("#embarkedheading i").addClass("fa-minus-circle");
    }
    if ($("#embarkedheading").attr("aria-expanded") == "true") {
        $("#embarkedheading i").removeClass("fa-minus-circle");
        $("#embarkedheading i").addClass("fa-plus-circle");
    }
});
```

Figure 1 | Handle Instruction Modal Collapse Icons

SLIDE

Handle Result Modal

- **Step 01: we will display the result modal only when the predicted result will be received in modal**

Javascript Code – Handle Result Modal

```
// Handling of Result modal to be displayed
window.onload = function () {
    if (document.getElementById("result").innerText) {
        $("#resultmodal").modal("show");
    }
};
```

Figure 2 | Handle Result Modal

SLIDE

Handle Input Control Button

- **Step 01: We have to control the content of Input Control Button text to switch between view/hide inputs and change the color accordingly**
  - **By default the button will be of green color with text Click to view inputs and when the inputs are hidden it will remain as above**
  - **When the inputs will be displayed the button will be of gray color with the text Click to minimize inputs**

Javascript Code – Handle Input Control Button

```
// Handle Display of inputs on result modal
$("#inputhandler").click(function () {
    if ($("#inputhandler").attr("aria-expanded") == "false") {
        document.getElementById("inputhandler").innerText =
            "Click to minimize inputs";
        $("#inputhandler").removeClass("btn-success");
        $("#inputhandler").addClass("btn-secondary");
    }
    if ($("#inputhandler").attr("aria-expanded") == "true") {
        document.getElementById("inputhandler").innerText =
            "Click to view inputs";
        $("#inputhandler").removeClass("btn-secondary");
        $("#inputhandler").addClass("btn-success");
    }
});
```

Figure 3 | Handle Input Control Button

SLIDE

Handle Predicted Result

- **Step 04: In this part we will control size and color for predicted result**
  - **Green color when predicted result is Survived**
  - **Red Color when predicted result is Not Survived**

Javascript Code – Handle Predicted Result

```
$(document).ready(function () {
    // Handle Predicted Result
    $("#result").ready(function () {

        // Handling the predited result to be green in case of ** Survived **
        if ($("#result").text() == "Survived") {
            document.getElementById("result").innerHTML =
                '<h1 id="result"><strong> Survived </strong></h1> ';
```

```
            $("#result").css("color", "green");
            document.getElementById("result").style.display = "block";
        }

        // Handling the predited result to be red in case of ** Not Survived *
*
        if ($("#result").text() == "Not Survived") {
            document.getElementById("result").innerHTML =
                '<h1 id="result"><strong> Not Survived </strong></h1>';
            $("#result").css("color", "#f22849");
            document.getElementById("result").style.display = "block";
        }
    });
});
```

Figure 4 | Handle Predicted Result

Make Application Routing Handler File

SLIDE

Make Application Routing Handler File

- **Step 1: Open any IDE to create app.py file that will handle the routing process of the application**
  - **In my case I am using Visual Studio Code 1.49.2**
- **Step 2: Click on Open Folder**

- **Popular IDE are:**
  - **Visual Studio Code 1.49.2**
  - **Brackets 1.14.2**
  - **Notepad++ 7.8.7**

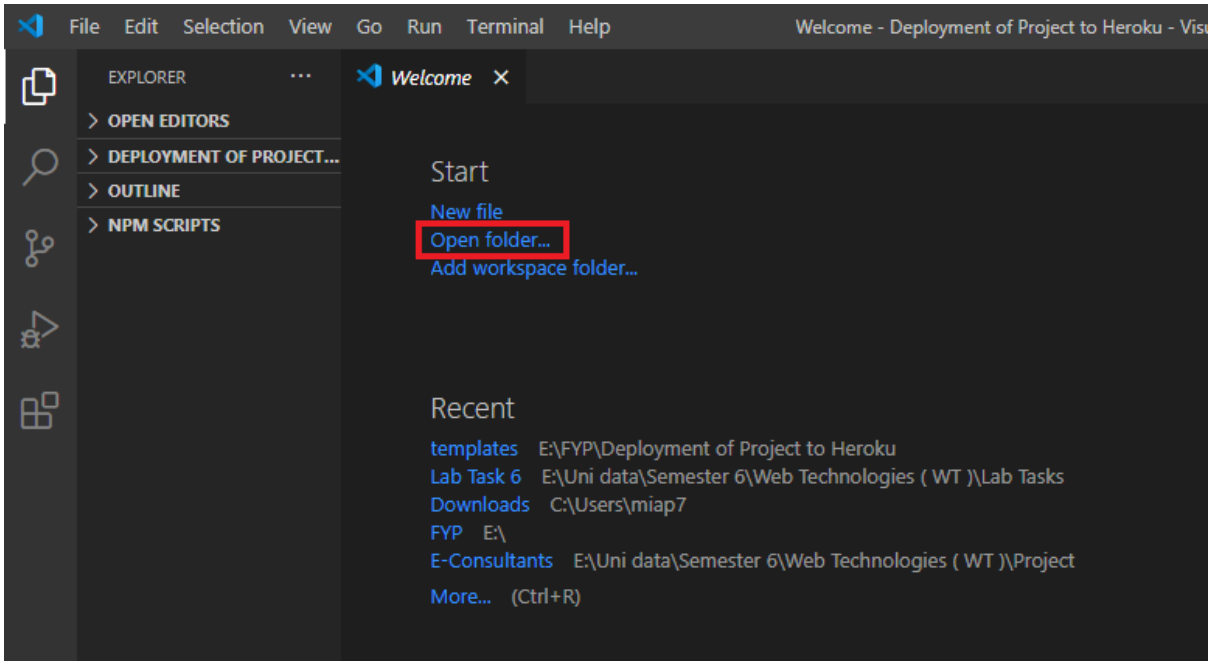

Figure 1 | Make Application Routing Handler File

SLIDE

Make Application Routing Handler File

- **Step 2: A new wizard will be displayed. Select OS (C:) from the left menu and then select your project folder and finally click on Select Folder below**
- **In my case the project folder name is Model Deployment**

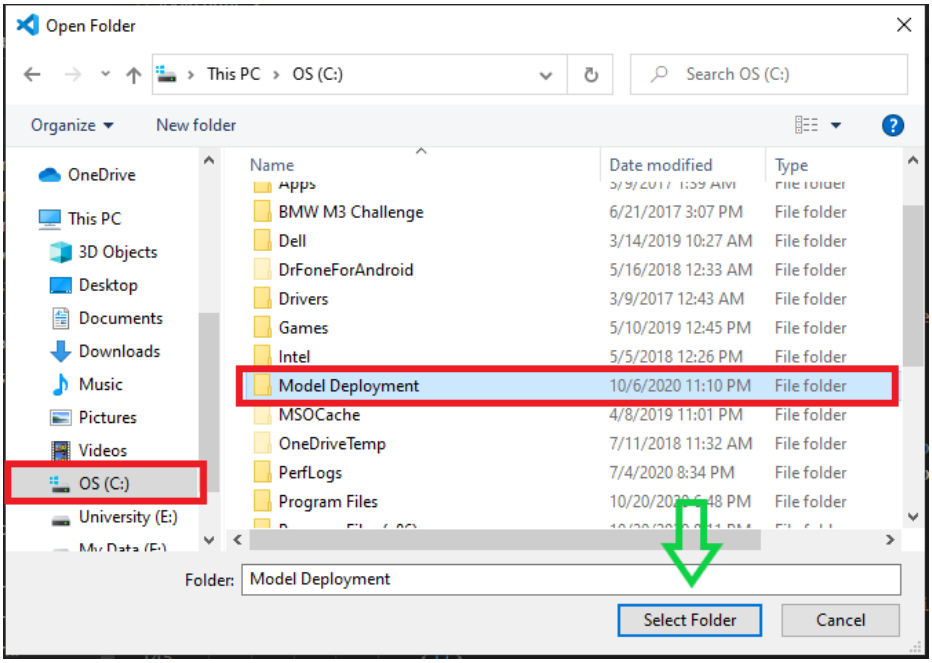

Figure 2 | Make Application Routing Handler File

SLIDE
Make Application Routing Handler File

- **Step 3: Your folder will be loaded in Visual Studio Code. You will see the below interface where you need to click  to create new file and rename that file to app.py**

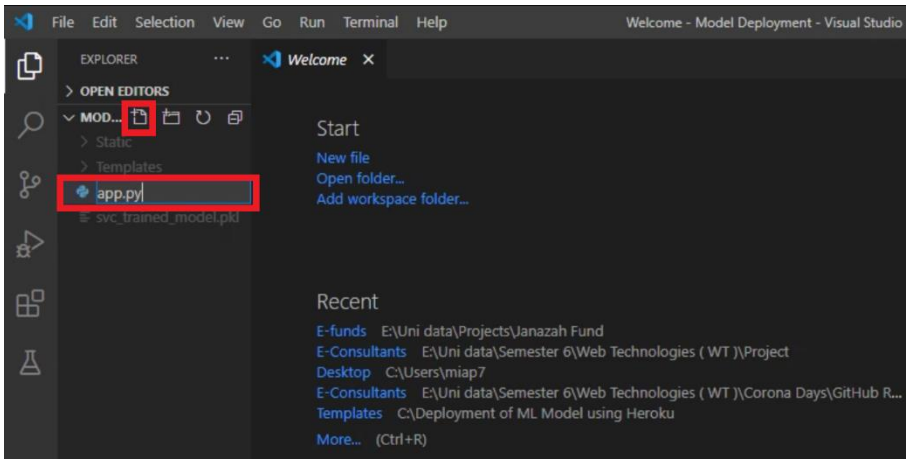

Figure 3 | Make Application Routing Handler File

SLIDE
Make Application Routing Handler File Cont…

- **Below code is according to the model in my case Titanic Passenger Survival Predictor**
- **Step 4: Import libraries which will be used in your app.py to handle different routes**

```
# import libraries
import numpy as np
from flask import Flask, request, jsonify, render_template
import pickle
```

Figure 4 | Make Application Routing Handler File

SLIDE

Make Application Routing Handler File Cont…

- **Step 5: Copy your trained model from your project folder and paste it in the current folder**
- **Step 6: Create a flask app named as app and load the trained model**
- **In my case the the model name is**
    - **svc_trained_model.pkl**

```
# create app and load the trained Model
app = Flask(__name__)
model = pickle.load(open('svc_trained_model.pkl', 'rb'))
```

Figure 5 | Make Application Routing Handler File

SLIDE

Make Application Routing Handler File Cont…

- **Step 7: Copy your Front-End Interface File (Already Built before) and paste it into the Templates folder inside your current Folder**
- **In my case the interface file is index.html**
- **Step 8: In Home route below the html file is rendered**

```
# Route to handle HOME
@app.route('/')
def home():
    return render_template('index.html')
```

Figure 6 | Make Application Routing Handler File

SLIDE

Make Application Routing Handler File Cont…

- **Step 9: In Predict route when this route will be hit with POST method firstly an array will be declared to save inputs and then the inputs from Front-End Interface will be saved into that array**
- **Afterwards the the trained model will predict the result by getting inputs array as parameter**
- **The final result will be predicted and stored in prediction**

- **Then interface file will be loaded again with predicted result passed to that file**

NOTE : In HTML file {{predicted_result}} must be present to load the result

```
# Route to handle PREDICTED RESULT
@app.route('/predict',methods=['POST'])
def predict():

    inputs = [] # declaring input array

    inputs.append(request.form['pclass'])
    inputs.append(request.form['gender'])
    inputs.append(request.form['siblings'])
    inputs.append(request.form['embarked'])

    final_inputs = [np.array(inputs)]
    prediction = model.predict(final_inputs)

    if(prediction[0] == 1):
        return render_template('index.html', predicted_result = 'Survived')
    if(prediction[0] == 0):
        return render_template('index.html',  predicted_result = 'Not Survived')
```

Figure 7 | Make Application Routing Handler File

SLIDE

Make Application Routing Handler File

- **Step 10: In the end the flask app needs to be loaded**

```
if __name__ == "__main__":
    app.run(debug=True)
```

Figure 8 | Make Application Routing Handler File

Make Procfile

SLIDE
Make Procfile

- **Step 1: In Visual Studio again click  to create new file and rename that file to Procfile**

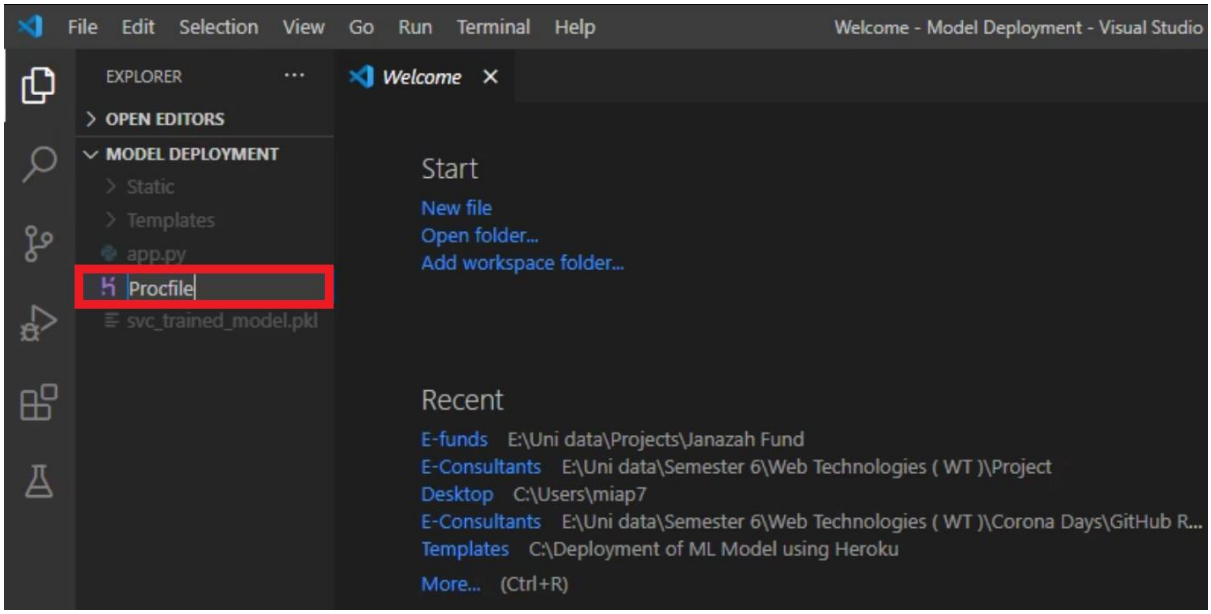

Figure 1 | Make Procfile

SLIDE
Make Procfile

- **Step 2: In Procfile type the following code with the same syntax below.**
- **Here the first app is the name of application routing handler file which have used as app.py and the second app is the name of flask app which we have used as app in application routing handler file**

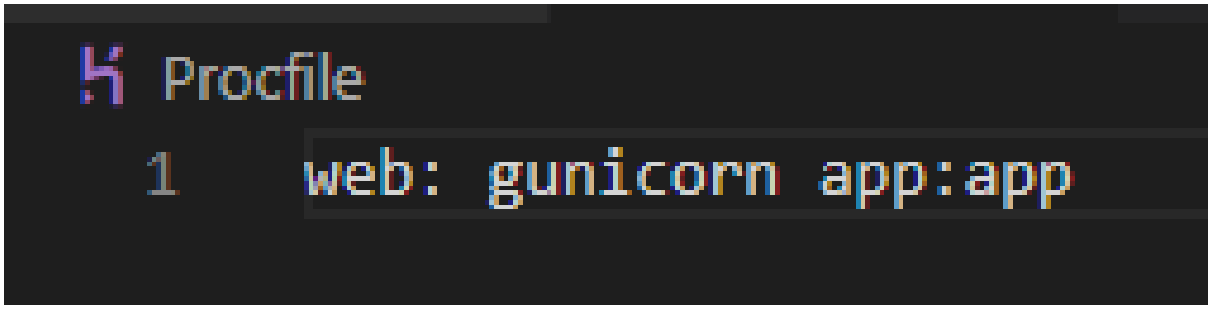

Figure 2 | Make Procfile

Make Requirements.txt File

SLIDE
Make Requirements.txt File

- **Step 1: Click on the Start button in the Task Bar**

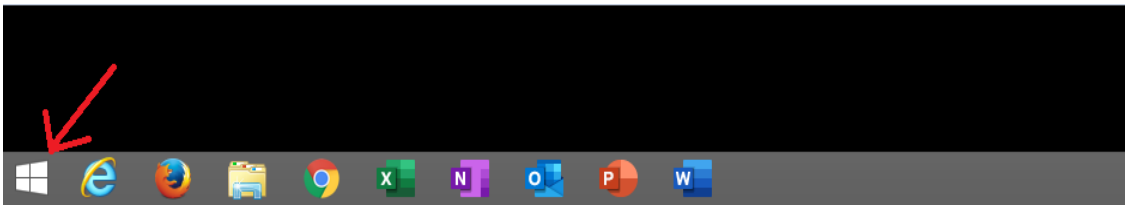

Figure 1 | Make Requirements.txt File

SLIDE
Make Requirements.txt File Cont…

- **Step 2: Search for Command Line Prompt (CMD)**
- **If you are using Windows 8/8.1 Pro, Search button will appear as shown below**

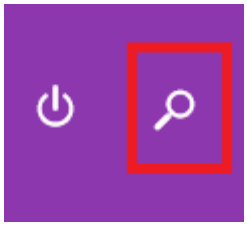

Figure 2 | Make Requirements.txt File

OR

SLIDE
Make Requirements.txt File Cont…

- **Step 2: Search for Command Line Prompt (CMD)**
- **If you are using Windows 10 , Search button will appear as shown in below figure**

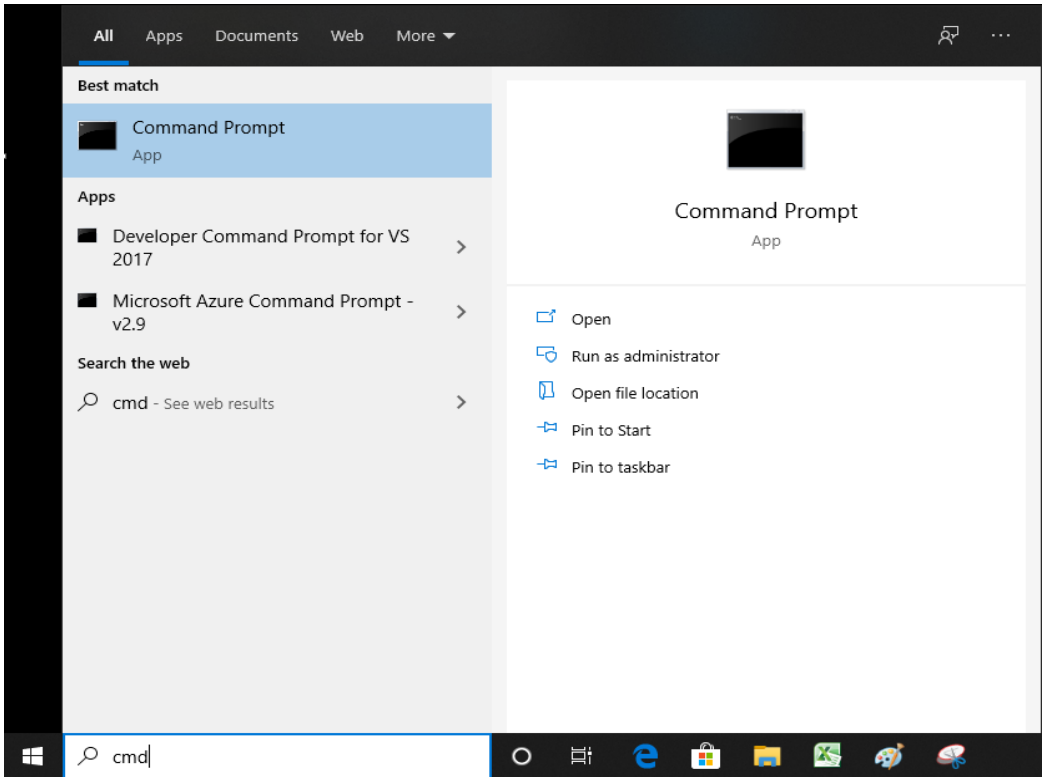

Figure 3 | Make Requirements.txt File

SLIDE
Make Requirements.txt File Cont…

- **Step 3: Click on Command Prompt**

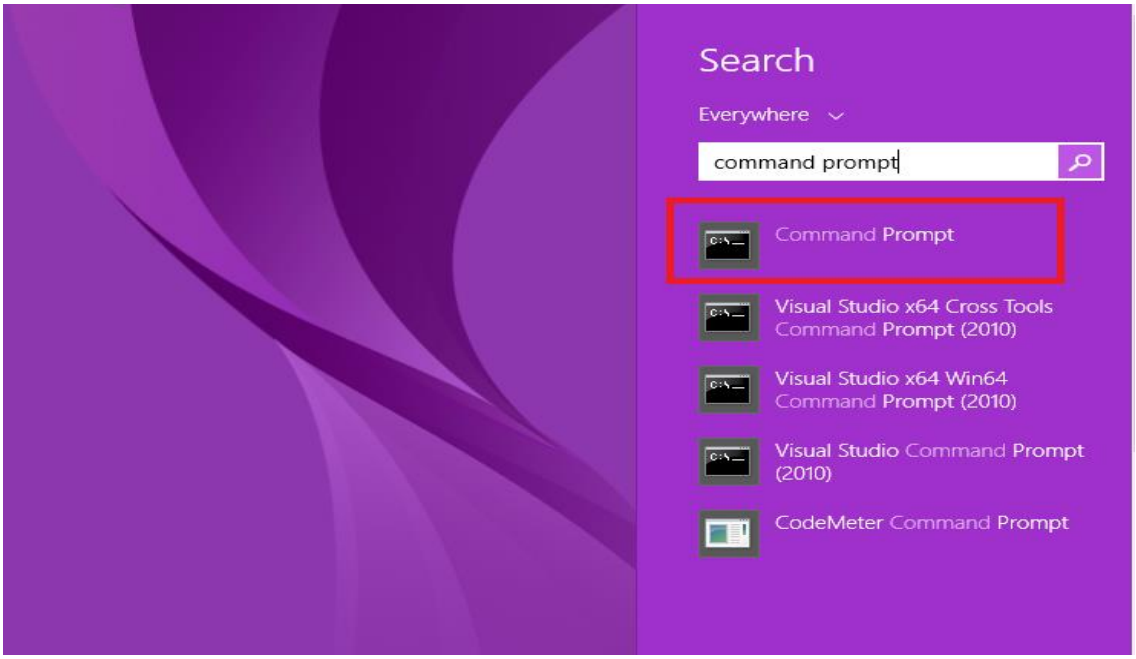

Figure 4 | Make Requirements.txt File

SLIDE

Make Requirements.txt File Cont…

- **Command Prompt window will appear on your Computer Screen**
- **Step 4: Change the path to your project folder path**
- **In my case the project folder is Titanic Project in OS (C:) Drive**

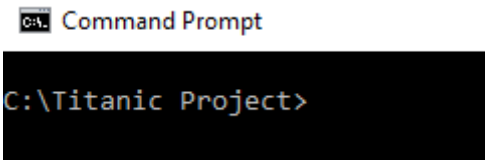

Figure 5 | Make Requirements.txt File

SLIDE

Make Requirements.txt File Cont…

- **Step 5: Type pip freeze > requirements.txt in command prompt**

Command
pip freeze > requirements.txt

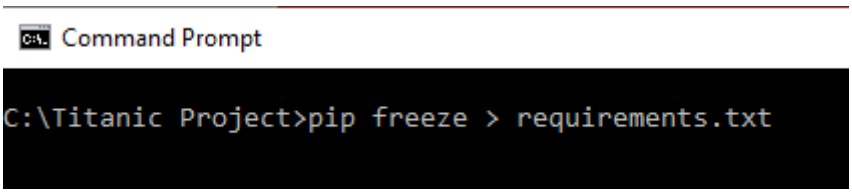

Figure 5 | Make Requirements.txt File

SLIDE

Make Requirements.txt File Cont…

- After execution of this command requirements.txt file will be created in your project folder containing all the pacakges you have installed while using your model
- In my case the requirements.txt is as below

Figure 5 | Make Requirements.txt File

Uploading Code to Gihub

SLIDE

Uploading Code to Gihub

- **Step 1: Open your Web Browser and type https://github.com and press Enter**
- **I am using Google Chrome 85.0.4183.102**
- **Github site will be loaded in your browser**
- **Step 2: Fill out the signup form (If your account doesnot exist) and click on Sign up for GitHub button**
- **If you have already created your account on github click Signin on top navbar**

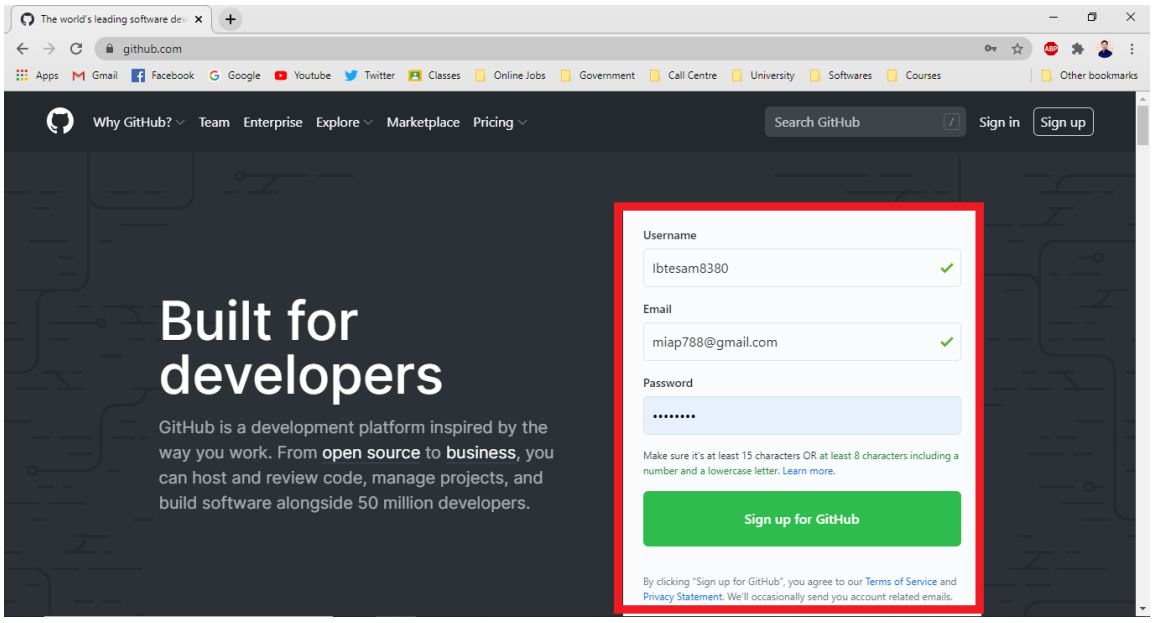

Figure 1 | Uploading Code to Gihub

SLIDE

Uploading Code to Gihub Cont…

- **New Interface with heading Create your account will be loaded in your browser**
- **Step 3: Solve the Captcha and click on Join a free plan button**

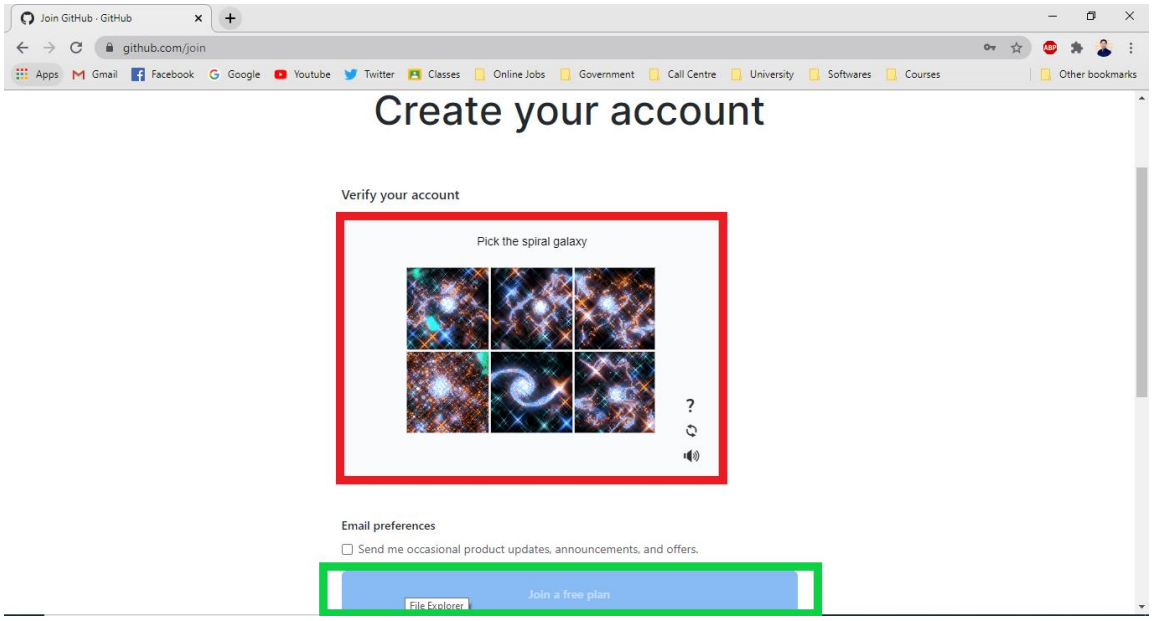

Figure 2 | Uploading Code to Gihub

SLIDE

Uploading Code to Gihub Cont…

- **New Interface with heading Welcome to GitHub will be loaded in your browser**
- **Step 4: Answer the questions given in the interface (as shown in figure below) by clicking on your suitable option**

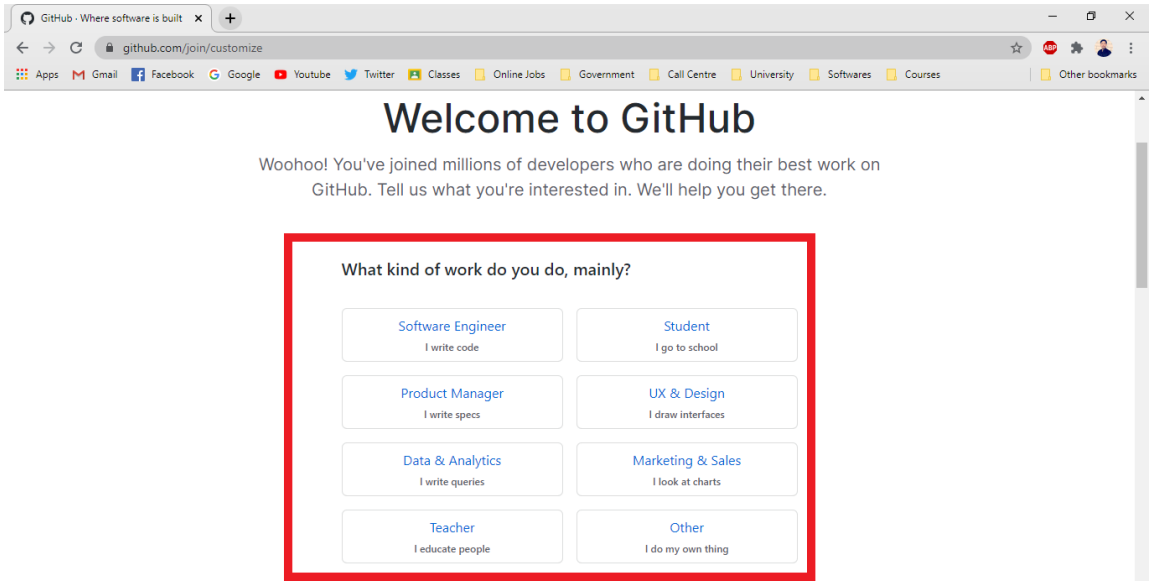

Figure 3 | Uploading Code to Gihub

SLIDE

Uploading Code to Gihub Cont…

- **New Interface with heading Please verify your email address will be loaded in your browser**

- **Step 5: Login to your email account and click on the verification link sent to you from GitHub**

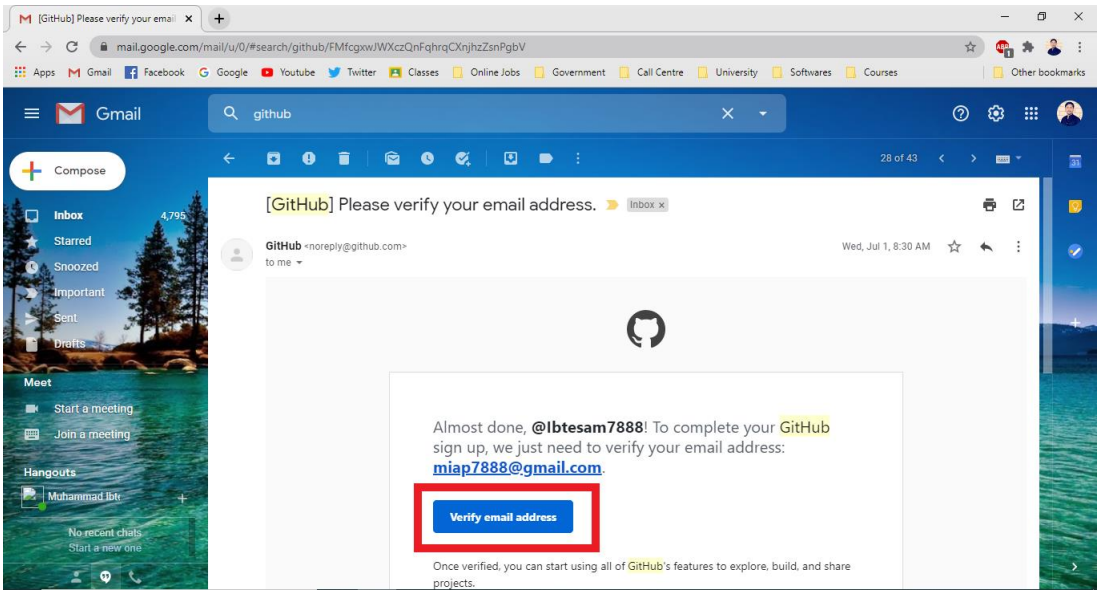

Figure 4 | Uploading Code to Gihub

SLIDE

Uploading Code to Gihub Cont…

- **If you have not received email from GitHub come back to github and click on Resend verification email button**

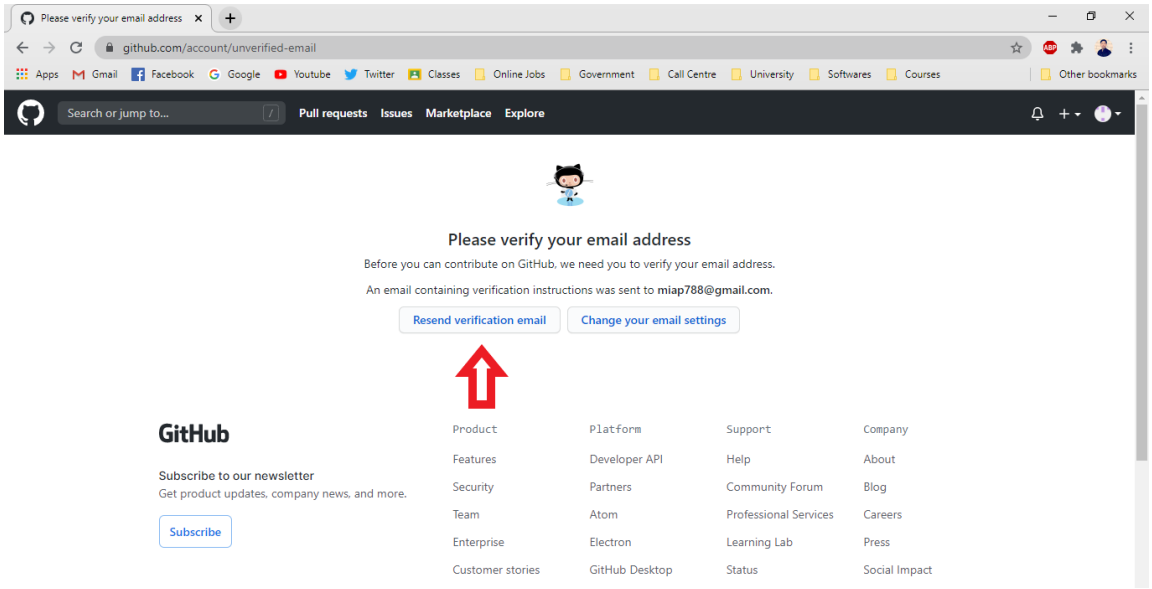

Figure 5 | Uploading Code to Gihub

SLIDE

Uploading Code to Gihub Cont…

- **Dashboard will loaded in your browser screen**
- **Step 6: Click on New button to create new repository**

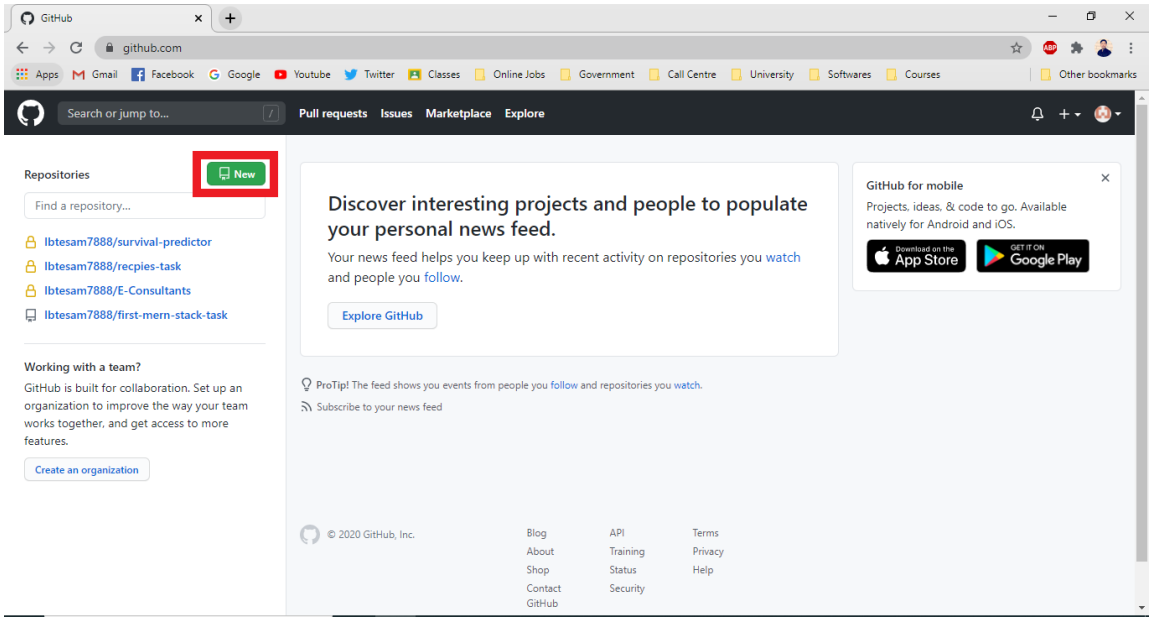

Figure 6 | Uploading Code to Gihub

SLIDE

Uploading Code to Gihub Cont…

- **New Repository Form will be loaded in your browser screen.**
- **Step 7: Enter any unique name of the repository in the form, change the privacy if needed and click on Create Repository button**
- **In my case the repository name is ml-project-deployment**

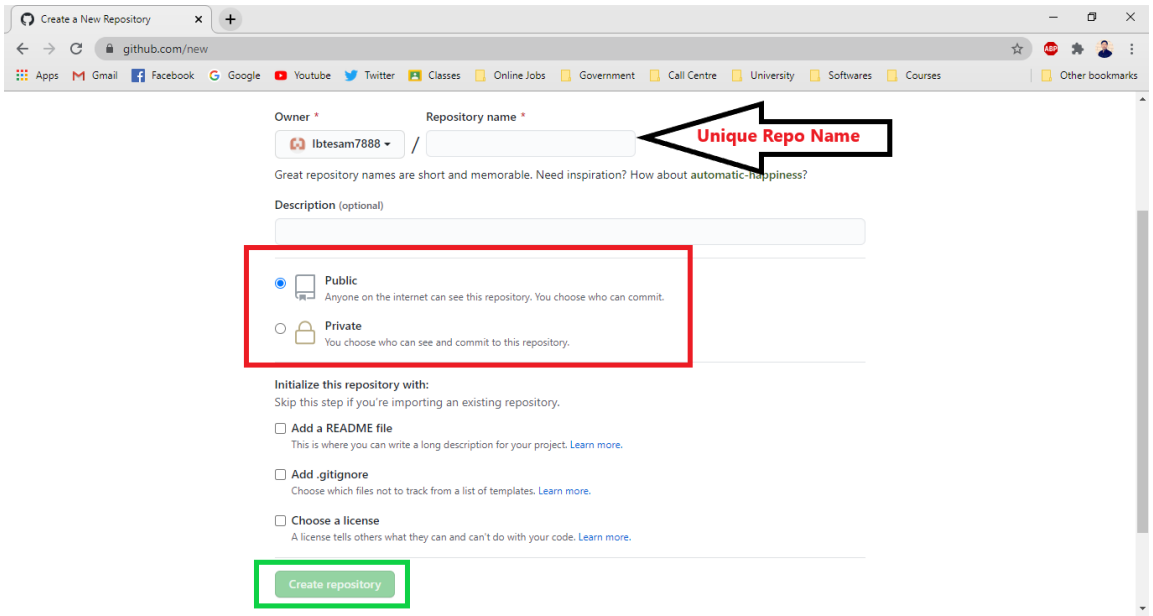

Figure 7 | Uploading Code to Gihub

SLIDE

Uploading Code to Gihub Cont…

- **Step 8: Click on upload an existing file link to upload the already built files from our system.**
- **Note that files can be uploaded using Command Line Interface (CLI) but in this case we are using Graphical User Interface (GUI)**

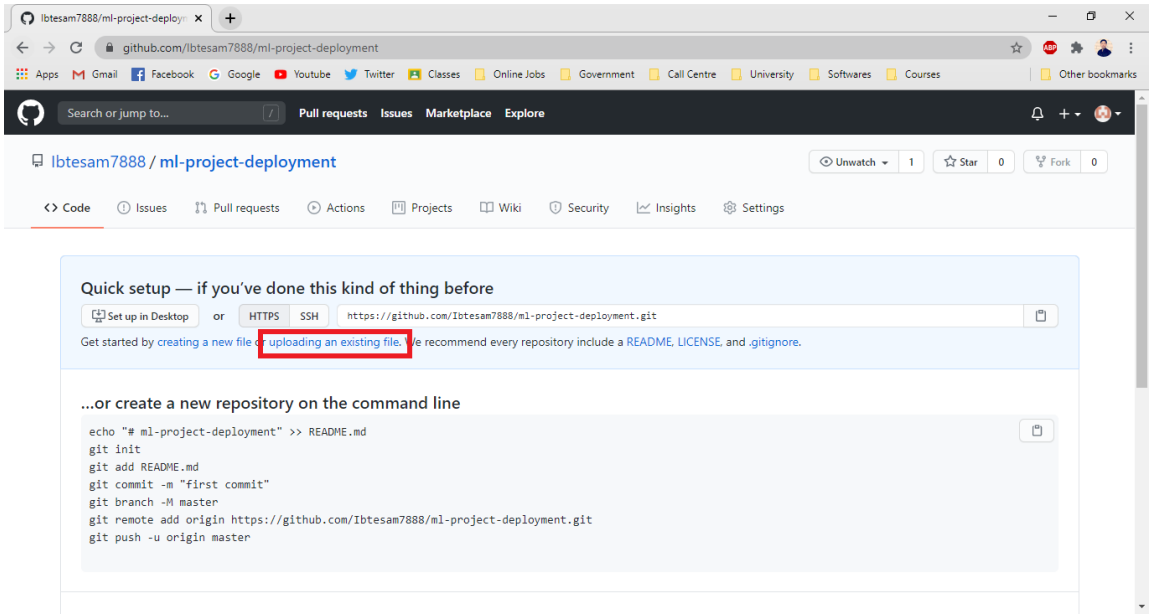

Figure 8 | Uploading Code to Gihub

SLIDE

Uploading Code to Gihub Cont…

- **Step 9: Drag and Drop all the files (including the Templates folder) from your deployment folder**
- **In my case the folder is named as Model Deployment**

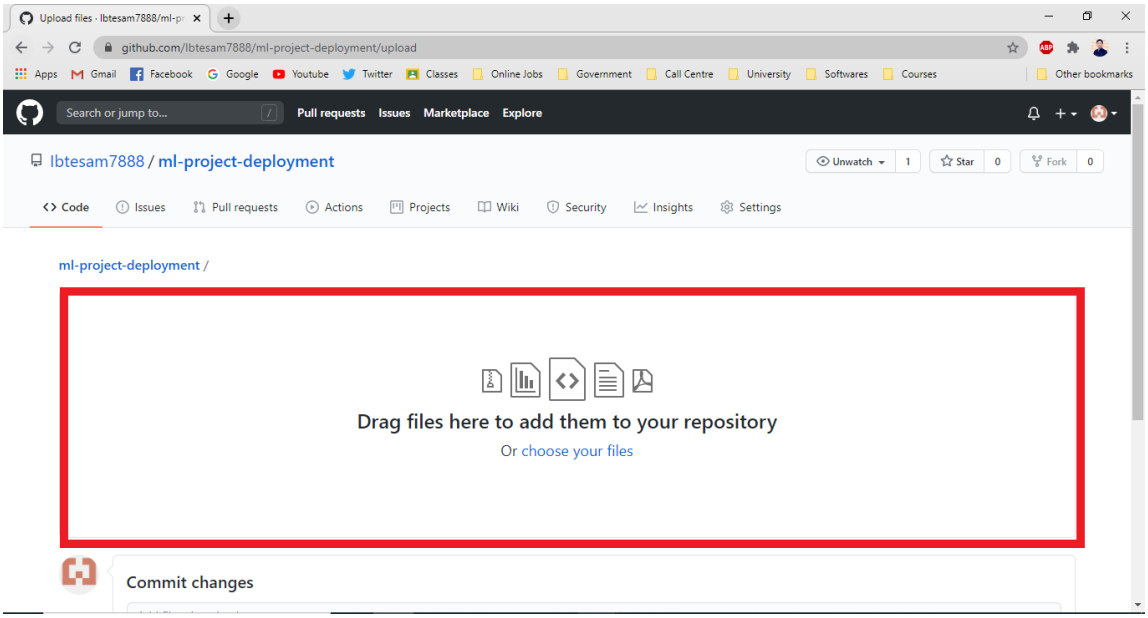

Figure 9 | Uploading Code to Gihub

SLIDE
Uploading Code to Gihub Cont…

- **Step 10: After successful uploading of all the files (as sown in figure below) name the commit in Commit Changes section and click on Commit Changes button**
- **In my case the commit name is First Commit**

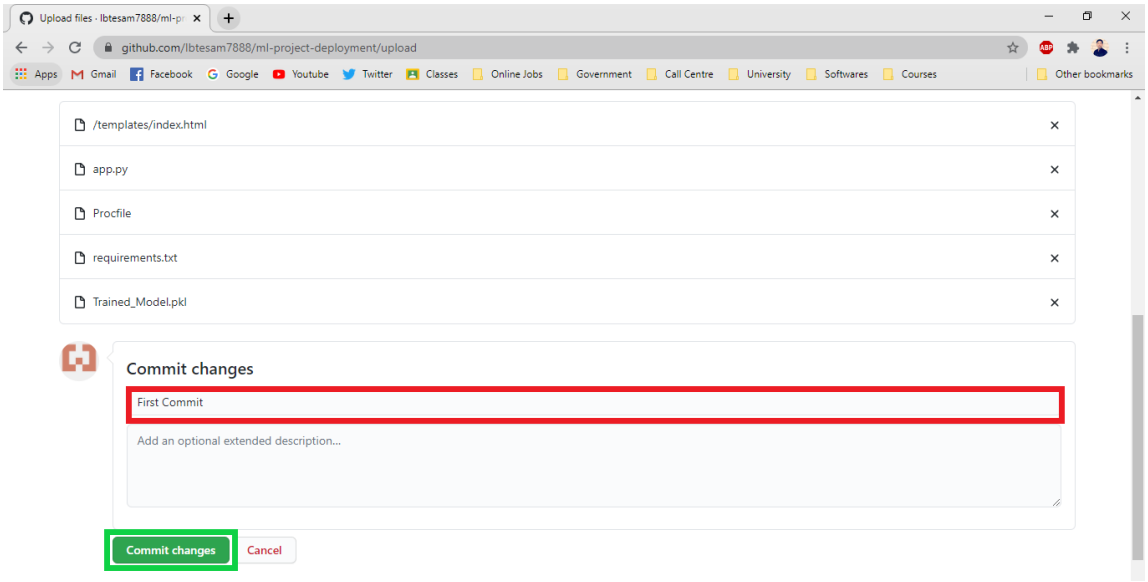

Figure 10 | Uploading Code to Gihub

SLIDE

Uploading Code to Gihub

- **If you see the template given below in your browser screen, then Alhamdulillah!!!! you are all over with GitHub**

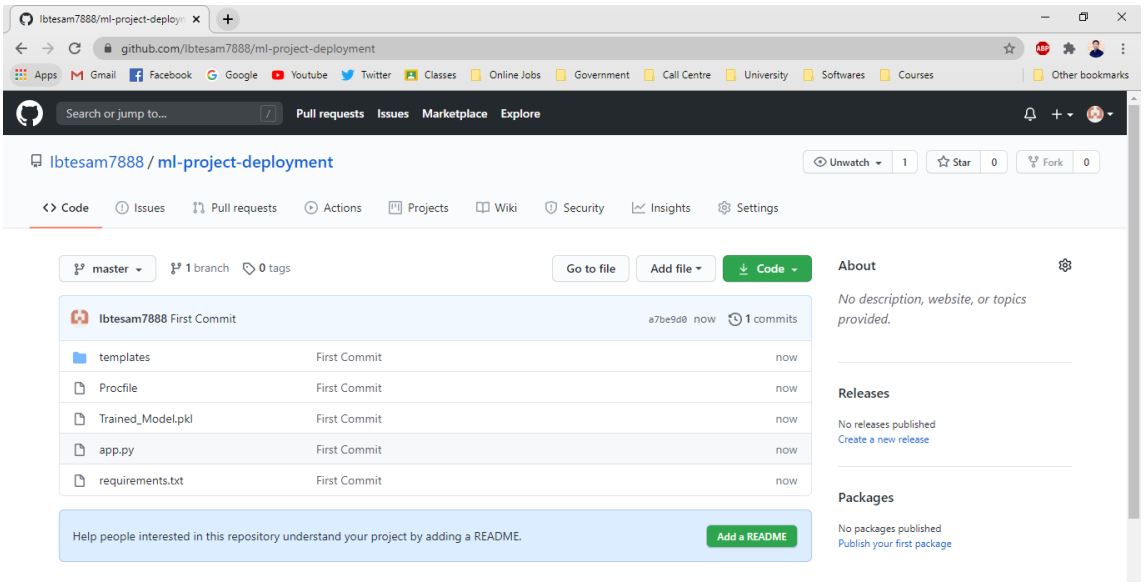

Figure 11 | Uploading Code to Gihub

Deplyment using Heroku

SLIDE

Deployment using Heroku

- **Step 1: Open your Web Browser and type https://heroku.com and press Enter**
- **I am using Google Chrome 85.0.4183.102**
- **Heroku site will be loaded in your browser screen**
- **Step 2: Click on Signup button to create new account on heroku**
- **If you have already created your account on heroku click Login on top navbar**

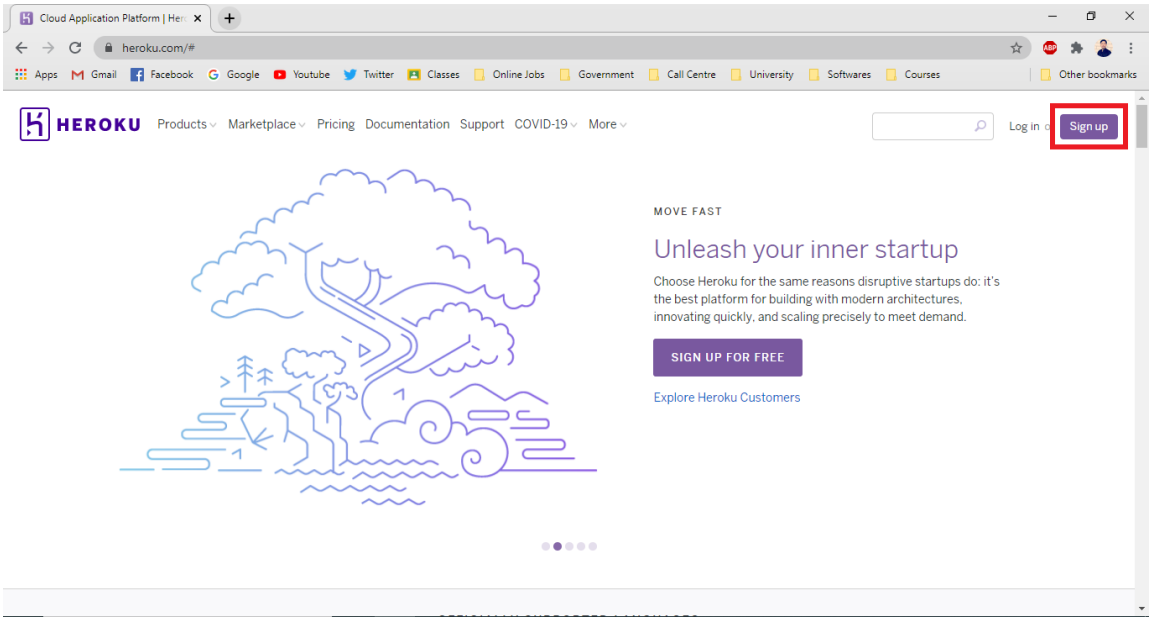

Figure 1 | Deployment using Heroku

SLIDE

Deployment using Heroku Cont…

- **Step 3: Fill out the signup form by writing your valid FirstName, LastName, Email and selecting your suitable Role, Country and Primary Development Language**
- **Step 4: Solve the Captcha and click on CREATE FREE ACCOUNT button**

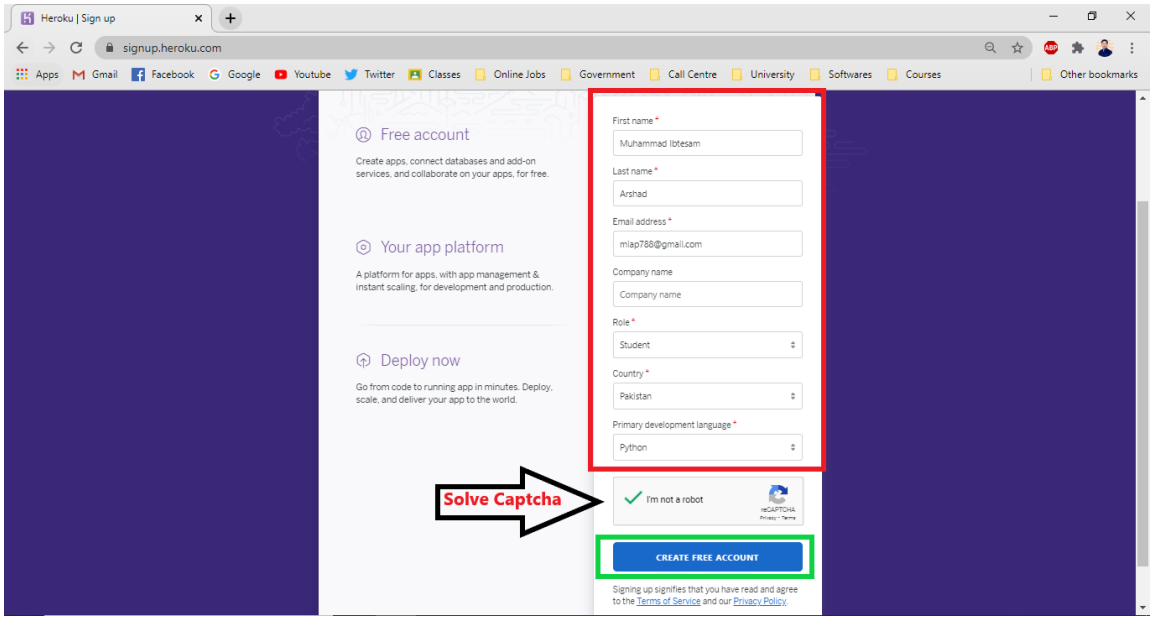

Figure 2 | Deployment using Heroku

SLIDE

Deployment using Heroku Cont…

- **An interface will be loaded in your browser screen with the message to verify your email**
- **Step 5: Login to your email account and click on the verification link sent to you by Heroku**

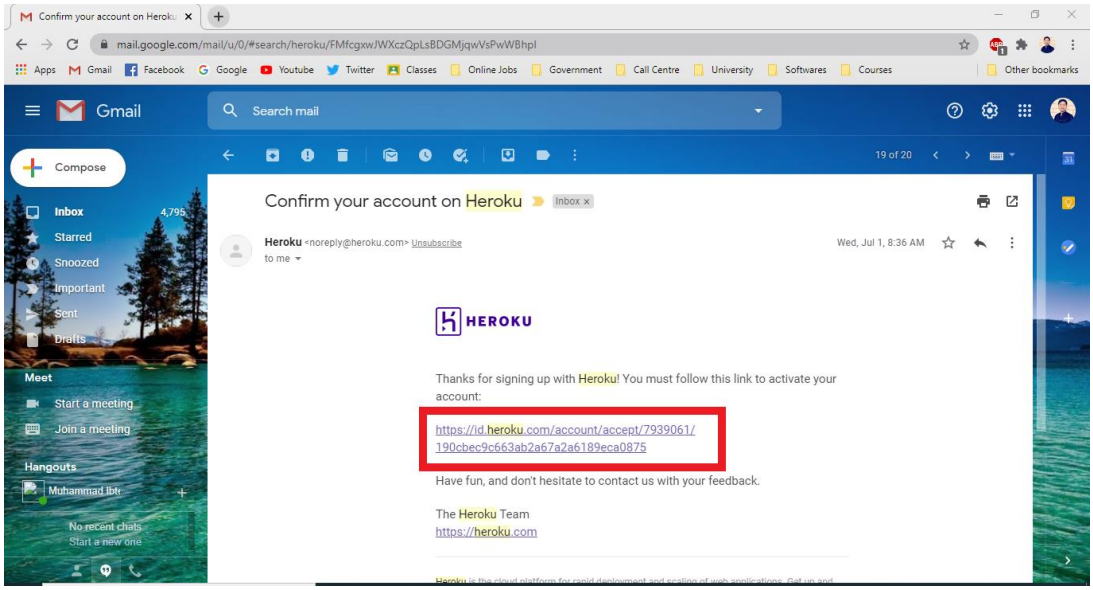

Figure 3 | Deployment using Heroku

SLIDE

Deployment using Heroku Cont…

- **If you didn't receive email from Heroku check your email address or check spam mails**

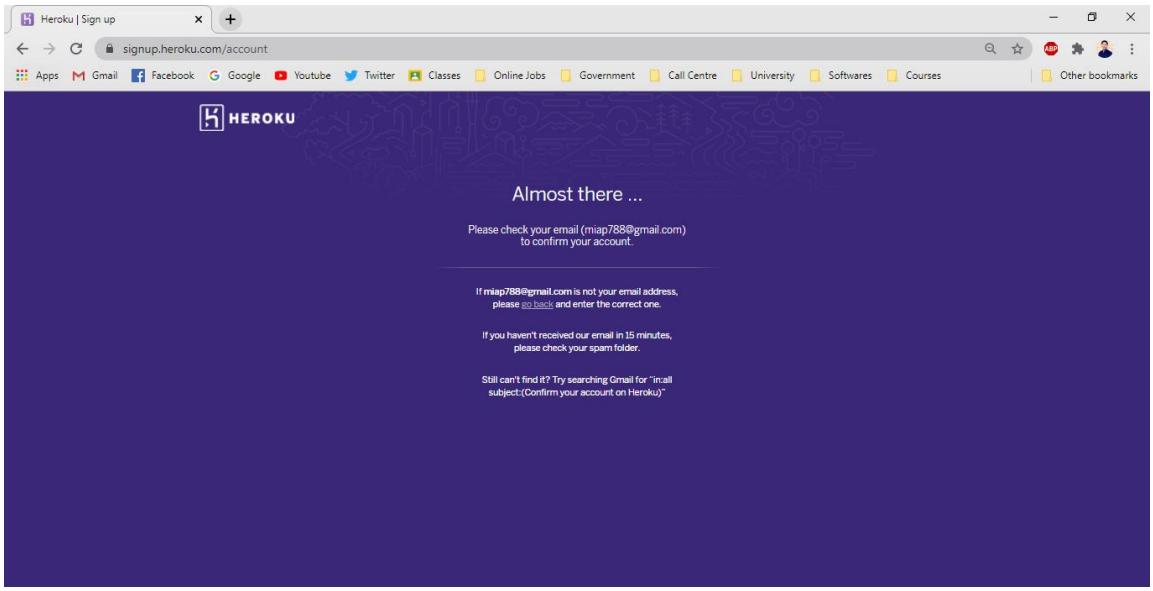

Figure 4 | Deployment using Heroku

SLIDE

Deployment using Heroku Cont…

- **Step 6: After successful verification, Login to your Heroku account by providing your credentials and click on Log In button**

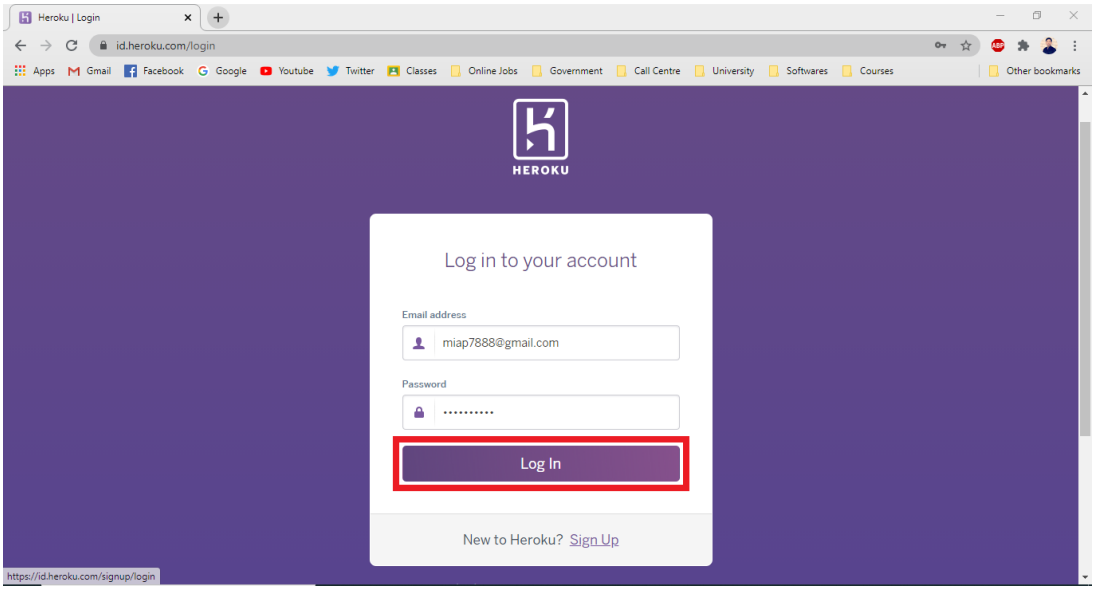

Figure 5 | Deployment using Heroku

SLIDE

Deployment using Heroku Cont…

- **Heroku Dashboard will be loaded in your browser screen**
- **Step 7: Click on New button on the top right corner and a dropdown menu will appear where you need to click on Create New app link**

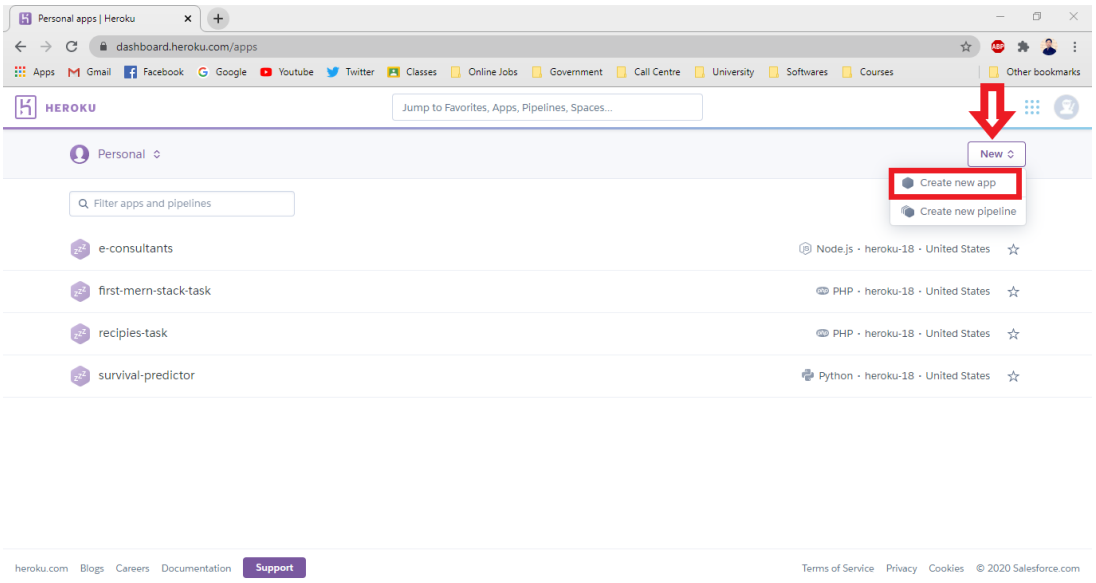

Figure 6 | Deployment using Heroku

SLIDE

Deployment using Heroku Cont…

- **Step 8: Provide a unique name for your app in App Name input field and click on Create app button**
- **In my case the app name is ml-project-deployment**

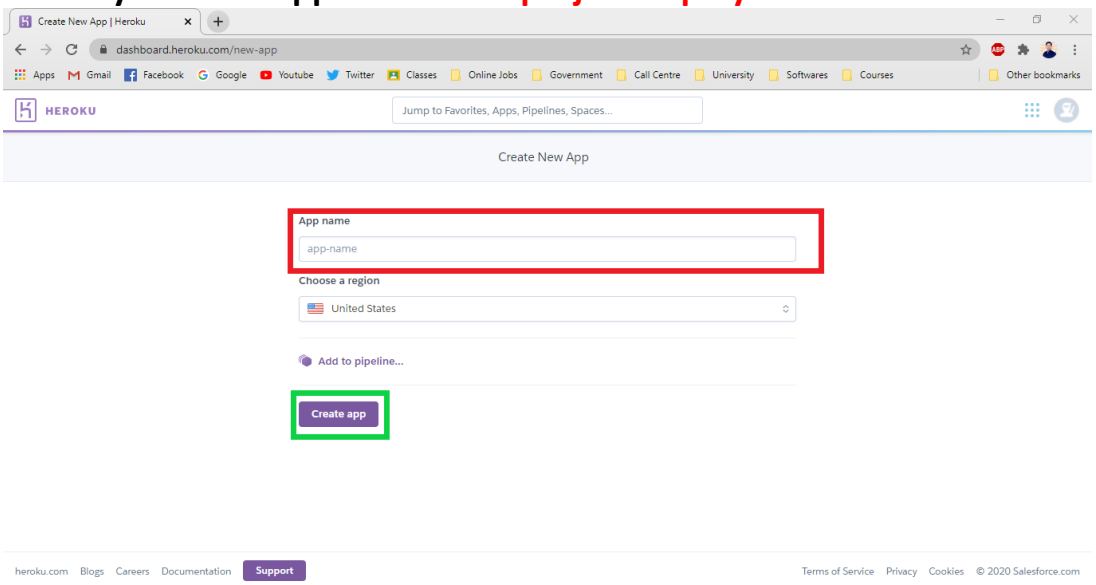

Figure 7 | Deployment using Heroku

SLIDE

Deployment using Heroku Cont…

- **Step 9: After successful creation of your app, Click on Connect to GitHub tab**

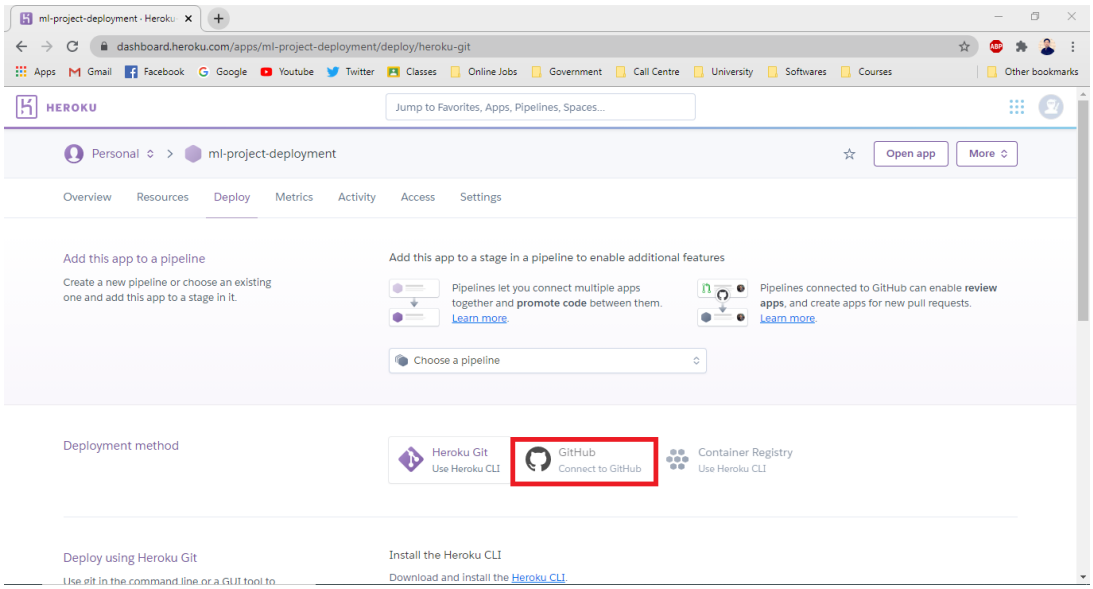

Figure 8 | Deployment using Heroku

SLIDE

Deployment using Heroku Cont…

- **Step 10: Search your GitHub repository by providing the correct name of GitHub repository and clicking on Search button**
- **Step 11: Your GitHub repository will be displayed below where you need to click on Connect button**

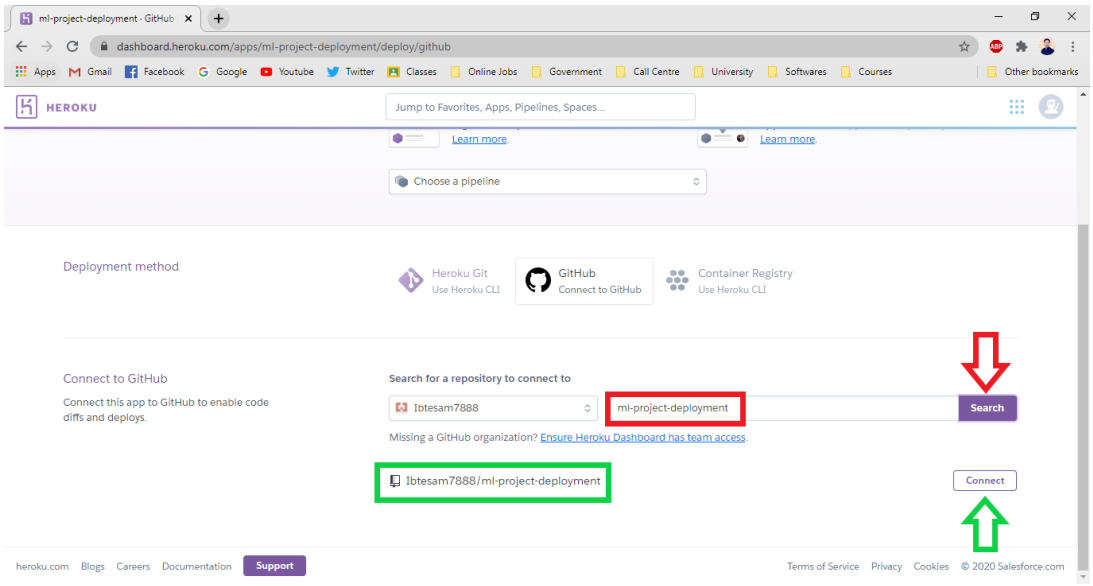

Figure 9 | Deployment using Heroku

SLIDE

Deployment using Heroku Cont…

- **Step 12: Click on Enable Automatic Deploys button and after that click on Deploy Branch button below**

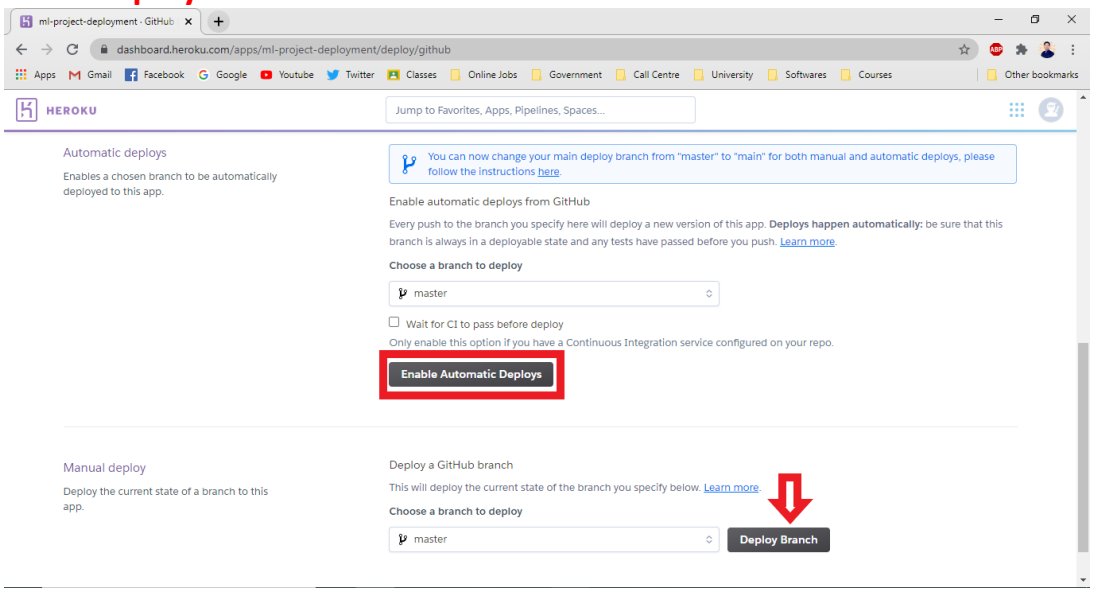

Figure 10 | Deployment using Heroku

SLIDE

Deployment using Heroku Cont…

- **Step 12: The build process will be started and heroku will automatically start downloading the pacakges listed in requirements.txt file**
- **The log will be displayed. If you find any error, fix that error and again deploy the branch**
- **After successful build of your application the message Your app was successfully deployed will be displayed below and a button View will also appear**
- **Step 13: Click on View button to check your application deployed**

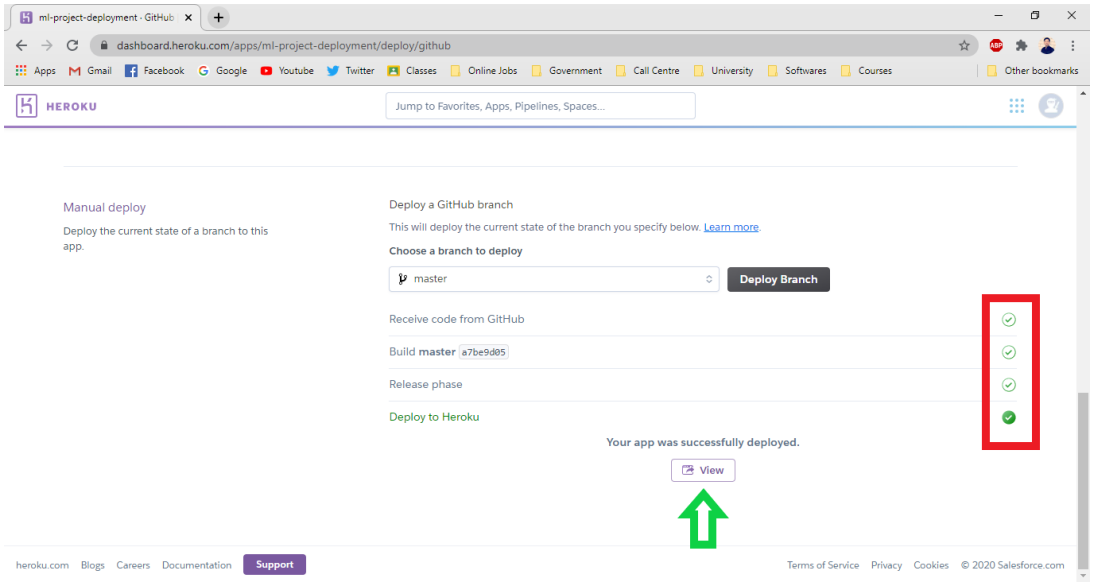

Figure 11 | Deployment using Heroku

SLIDE

Deployment using Heroku

- **Alhamdulillah!!!!! Your model has successfully deployed using heroku and will be live with your heroku application name along with heroku domain**
- **In my case the url is https://ml-project-deployment.herokuapp.com**

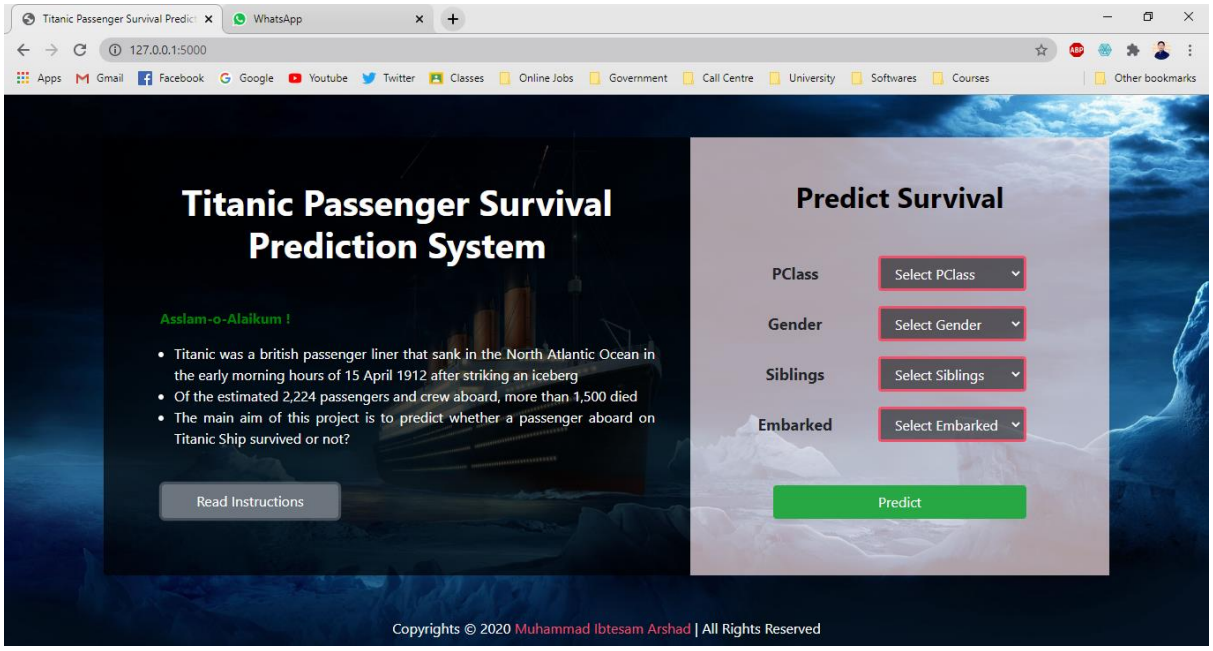

Figure 12 | Deployment using Heroku

Its Inspirational Quotes Time

Quote No 01

بدی چُھپ کر کرتے ہو نیکی بھی چُھپ کر کرو یہی اِخلاص ہے گناہ پر ندامت گناہ کو مٹا دیتی ہے
نیکی پر غرور نیکی کو تباہ کر دیتی ہے

حضرت علی رضی اللہ تعالی عنہ

Quote No 02

کسی کو اچھے عمل سے دلی خوشی دینا ہزار سجدے کرنے سے بہتر ہے

شیخ سعدی رحمت اللہ علیہ

Quote No 03

کتنا خوبصورت احساس ہے کہ جب ہم ''اللہ'' سے دِل کی بات کہتے ہیں
تو وہ سب کچھ جانتے ہوئے بھی ہماری بات سننا چاہتا ہے

اشفاق اَحمد

Quote No 04

اراستو سے کسی نے کہا
میں نے ایک معتبر شخص سے تمھارے بارے میں کچھ غلط باتیں سنی ہیں
' اراستو نے جواب دیا' غیبت کرنے والا معتبر کیسے ہو گیا

Quote No 05

صبر کا گھونٹ دوسروں کو پلانا آسان ہے
خود پیتے ہوئے پتہ چلتا ہے کے ایک ایک قطرہ پینا کتنا بھاری پڑتا ہے

اشفاق اَحمد

Its Poetry Time

Ghazal No 01

یہ معجزہ بھی محبت کبھی دکھائے مجھے
کے سنگ تجھ پہ گرے اور زخم آئے مجھے

وہ میرا دوست ہے سارے جہاں کو ہے معلوم
دغا کرے وہ کسی سے تو شرم آئے مجھے

وہ مہربان ہے تو اقرار کیوں نہیں کرتا
وہ بدگمان ہے تو سو بار آزمائے مجھے

وہی تو سب سے زیادہ ہے نقطہ چین میرا
جو مسکرا کے ہمیشہ گلے لگائے مجھے

برنگ اوڈ ملے گی اسے میری خوشبو
وہ جب بھی چاہے بڑے شوق سے جلائے مجھے

میں اپنی ذات میں نیلام ہو رہا ہوں قتیل
غم حیات سے کہہ دو خرید لائے مجھے

قتیل شفائی

Its Jokes Time

Joke No 01

کل رات میسیج آیا تھا
جیسے 10 لوگوں کو فارورڈ کرنا تھا
ورنہ بھاری نقصان اٹھانا پڑتا
میں نے اگنور کر دیا

اور صبح ہی میرے 2 بسکٹ چائے میں ڈوب گئے

Joke No 02

ایک آدمی گلی میں جا رہا تھا کہ اچانک آواز آئی

" رک جاؤ "

آدمی رکا ہی تھا کہ اس کے آگے ایک اینٹ آ کر گری اِس نے اللہ کا شکر ادا کیا

" کچھ دن کے بعد وہ سڑک پار کرنے لگا تھا کے پِھر وہی آواز آئی

" خبر دَر! رک جاؤ"

وہ رک گیا اور اسی وقت ایک گاڑی اِس سے ایک انچ آگے سے گزر گئے

آدمی کو پچھلا واقعہ یاد آ گے اِس نے چلا کر پوچھا

"کون ہو تم؟"

آواز آئی

" تمھارا نگہبان فرشتہ "

آدمی ڈبڈبائی ہوئی آنکھیں لے کر بولا

"حضرت آپ میرے نکاح کے وقت کہاں تھے؟"

Joke No 03

ایک خر گوش روزانہ سبزی کی دکان پر جاتا اور پوچھتا

گاجر ہے؟

سبزی والے نے تنگ آ کر اُس کے دانت توڑ دیے اور کہا

آپ گاجر کھا کر دیکھا

اگلے دن خر گوش پِھر آیا اور بولا

گاجر کا حلوہ ہے...؟؟؟؟

Its Poetry Time

Ghazal No 01

کسی کا یوں تو ہوا کون عمر بھر پھر بھی
یہ حُسن عشق کا دھوکہ ہے سب مگر پھر بھی

ہزار بار زمانہ ادھر سے گزرا ہے
نئی نئی سی ہے کچھ تیری راہگزر پھر بھی

خوشا اشارۂ پیہم، زہے سکوتِ نظر
دراز ہو کے فسانہ ہے مختصر پھر بھی

جھپک رہی ہے زمان و مکان کی آنکھیں
مگر ہے قافلہ، آمادۂ سفر پھر بھی

شبِ فراق سے آگے ہے آج میری نظر
کہ کٹ ہی جائے گی یہ شامِ بے سحر پھر بھی

لپٹ گیا ترا دیوانہ گرچہ منزل سے
اُڑی اُڑی سی ہے کچھ خاکِ رہگزر پھر بھی

تری نگاہ سے بچنے میں عمر گزری ہے
اُتر گیا رگ جاں میں یہ نیشتر پھر بھی

غمِ فراق کے کُشتوں کا حشر کیا ہو گا

یہ شامِ ہجر تو ہو جائے گی سحر پھر بھی

فنا بھی ہو کے گرانبارِ حیات نہ پوچھ

اُٹھائے اُٹھ نہیں سکتا یہ دردِ سر پھر بھی

ستم کے رنگ ہیں ہر التفاتِ پنہاں میں

کرم نُما ہیں ترے زور سر بسر پھر بھی

خطا معاف! ترا عفو بھی ہے مثلِ سزا

تری سزا میں ہے اک شانِ در گزر پھر بھی

اگرچہ بیخودیِ عشق کو زمانہ ہوا

فراق، کرتی رہی کام وہ نظر پھر بھی

Stop Complaining! Stop Criticizing! Let's Start Contributing

SLIDE

A True Story

- **Here I am writing a true story of one of my Respected Teachers (Prof. Dr. Yaseen Iqbal**
  Department of Physics, University of Peshawar, Pakistan)

SLIDE

Story

- **In 1996, I was a PhD student at University of Sheffield, England. One day, I was having a walk with my friends. We saw an Old Lady picking up French Fries (potato chips) from the Foot Path. One of my friends, said to the Old Lady**
  - **Mam! Why are you picking these? It is a crowded place and you may get hurt.**
- **Old Lady replied**
  - **Gentleman! This is MY Country. If it is dirty. I feel dirty.**
- **Remember**
  - **There is nothing like**
    - **Big Contribution or**
    - **Small Contribution**
  - **Contribution is Contribution** 😊
- **Let's Start Contributing from Today**

- **To make this Beautiful World, more Beautiful** 😊

Its Inspirational Quotes Time

Quote No 01

علم بے عمل اور عمل بے علم سے پرہیز کرو۔

حکیم لقمان رحمت اللہ علیہ

Quote No 02

خاموشی غصے کا بہترین علاج ہے۔

حضرت عثمان غنی رضی اللہ تعالی عنہ

Quote No 03

جو تجھے تیرے عیب سے آگاہ کرے وہ تیرا دوست ہے۔

حضرت عمر رضی اللہ تعالی عنہ

Quote No 04

جو شخص اپنی قدر آپ نہیں کرتا کوئی دوسرا شخص بھی اس کی قدر نہیں پہچانتا۔

حضرت علی رضی اللہ تعالی عنہ

Quote No 05

> سب سے بہتر وہ لقمہ ہے جسے اپنی محنت سے حاصل کیا جائے۔
>
> حضرت ابو بکر رضی اللہ تعالیٰ عنہ

SLIDE

Lecture Summary

SLIDE
Lecture Summary

- **To systematically perform any Real-world Task using a Template-based Approach, follow the following steps**
  - **Step 1: Completely and correctly understand the Real-world Task**
    - **Write down two main things**
      - **Given**
      - **Task**
  - **Step 2: Understand the Input and Output of the Real-world Task**
    - **Write down two main things**
      - **Input**
      - **Output**
  - **Step 3: Plan and Design a Template-based Approach to perform the Real-world Task**
    - **Step 3.1: Use Divide and Conquer Approach to break the Real-world Task into**
      - **Steps / Sub-steps / Sub-sub-steps**
    - **Step 3.2: For each Steps / Sub-steps / Sub-sub-steps**
      - **Check the Order and Flow between Steps / Sub-steps / Sub-sub-steps**
      - **Check the Connectivity and Independence between Steps / Sub-steps / Sub-sub-steps**
  - **Step 4: Use a Five Step Process to perform the Real-world Task**
    - **Step 4.1: Plan – in Mind**

        - **Step 4.2: Design – on Paper**
        - **Step 4.3: Execute – at Prototype level**
        - **Step 4.4: Execute – at Full Scale**
        - **Step 4.5: Take Feedback from Users / Audience and Domain Expert to further improve the solution of Real-world Task**
    - **Step 5: Document each and every Step, when performing a Real-world Task**
- **Alhumdullialh, in this Lecture we systematically learned (using a Template-based Approach) how to**
    - **Use the Best Trashing and Learning Methodology of the World to systematically perform any Real-world Task using a Template-based Approach**
    - **Deploy Machine Learning Model using Heroku**
    - **Verify Deployment of Model**
    - **Start Contributing from Today** 😊