Objective:

The goal is to create a web application using Google Apps Script that allows users to retrieve specific details from a Google Sheet based on the provided criteria: Pool ID and Employee ID.

Requirements:

1. User Interface:
   - The web app should have a clean and user-friendly interface.
   - Include a "Select Your Pool" drop down with options for ID1-1 and ID2, populated dynamically from the available sheet tabs.
   - Provide an input field labeled "Enter EMP ID" for users to input the Employee ID.
   - Include a button (CTA) named "Get Details" to trigger the data retrieval process.

2. Data Retrieval:

Basic Data

When the user selects a Pool ID and enters an Employee ID, the web app should fetch and display the following details from the corresponding sheet for the entered EMI ID:

- First Name
- User Role
- Total Number of Goods (`NO_OF_GOODS`) for the given Employee ID
- Sum of Market Price (`MKT_PRICE`) for the given Employee ID
- Sum of Sold Price (`SOLD_PRICE`) for the given Employee ID
- Average of SELL%

Data Table:

Where

- "Sell" value corresponds to or falls within any of the specified percentages such as SELL1_Percentage, SELL2_Percentage, and so forth. If the "Sell" percentage does not align with any of these specified percentages, display the value as "NA."
- The commission is the corresponding value for which the "Sell" matches. For instance, if the "Sell" value aligns with Sell1_Percentage, then the commission would be the Sell1_Price.
- Rest other fields are as is populated for the given EMI ID

Category	NO_OF_GOODS	MKT_PRICE	SOLD_PRICE	SELL%	Sell	Commission
A1	10,522	55,581	52,479.58	94.42%		
A2 = EXM	3,168	12,796	8,377.54	65.47%		
D2			0.00			
DD3	881	3,305	2,523.37	76.35%		

D4= 23-AMX	2,554	10,528	5,878.84	55.84%		
E1			0.00			
E2	2,182	8,864	2,921.57	32.96%		
F2 = RMI	1,459	5,925	1,398.30	23.60%		

Data Table Calculator:

The **EST_SOLD_PRICE** should be editable, allowing users to input values. Based on the user input, the SELL% should automatically adjust, calculated as EST_SOLD_PRICE divided by MKT_PRICE. Consequently, the Sell value should dynamically change according to the modified SELL%, and the Commission should also adapt based on the updated Sell value.

Category	NO_OF_GOODS	MKT_PRICE	SOLD_PRICE	EST_SOLD_PRICE	SELL%	Sell	Commission
A1	10,522	55,581	52,479.58		94.42%		
A2 = EXM	3,168	12,796	8,377.54		65.47%		
D2			0.00				
DD3	881	3,305	2,523.37		76.35%		
D4= 23-AMX	2,554	10,528	5,878.84		55.84%		
E1			0.00				
E2	2,182	8,864	2,921.57		32.96%		
F2 = RMI	1,459	5,925	1,398.30		23.60%		

UI Interface like

Basic Data	Data Table
Data Table Calculator:	

My Code.gs

```
// Code.gs (Google Apps Script Code)
function doGet() {
  return HtmlService.createHtmlOutputFromFile('Page')
      .setTitle('Employee Details Web App')
      .setSandboxMode(HtmlService.SandboxMode.IFRAME);
}

function getData(poolId, empId) {
  try {
    var sheet =
SpreadsheetApp.getActiveSpreadsheet().getSheetByName(poolId);
    if (!sheet) {
      throw new Error('Invalid Pool ID');
    }

    var dataRange = sheet.getDataRange();
    var data = dataRange.getValues();

    // Assuming the headers are in the first row
    var headers = data[0];
    var empIdIndex = headers.indexOf('EMP_ID');
    var firstNameIndex = headers.indexOf('First Name');
    var userRoleIndex = headers.indexOf('User Role');
    var noOfGoodsIndex = headers.indexOf('NO_OF_GOODS');
    var mktPriceIndex = headers.indexOf('MKT_PRICE');
    var soldPriceIndex = headers.indexOf('SOLD_PRICE');
    var sellPercentageIndex = headers.indexOf('SELL%');

    var totalSellPercentage = 0; // Variable to store the total 'SELL%' for
calculating average
    var totalRows = 0;

    for (var i = 1; i < data.length; i++) {
      if (data[i][empIdIndex] == empId) {
        totalRows++;
        totalSellPercentage += parseFloat(data[i][sellPercentageIndex]) ||
0;
```

```
        return {
          firstName: data[i][firstNameIndex],
          userRole: data[i][userRoleIndex],
          noOfGoods: data[i][noOfGoodsIndex],
          mktPriceSum: data.reduce((sum, row) => sum +
(parseFloat(row[mktPriceIndex]) || 0), 0),
          soldPriceSum: data.reduce((sum, row) => sum +
(parseFloat(row[soldPriceIndex]) || 0), 0),
          sellPercentageAvg: totalSellPercentage / totalRows, // Calculate
average 'SELL%'
        };
      }
    }

    throw new Error('Employee ID not found');
  } catch (error) {
    return { error: error.message };
  }
}

function getEditableTableData(poolId, empId) {
  try {
    var sheet =
SpreadsheetApp.getActiveSpreadsheet().getSheetByName(poolId);
    if (!sheet) {
      throw new Error('Invalid Pool ID');
    }

    var dataRange = sheet.getDataRange();
    var data = dataRange.getValues();

    // Assuming the headers are in the first row
    var headers = data[0];
    var empIdIndex = headers.indexOf('EMP_ID');
    var categoryIndex = headers.indexOf('Category');
    var noOfGoodsIndex = headers.indexOf('NO_OF_GOODS');
    var mktPriceIndex = headers.indexOf('MKT_PRICE');
    var soldPriceIndex = headers.indexOf('SOLD_PRICE');
```

```
    var estSoldPriceIndex = headers.indexOf('EST_SOLD_PRICE');
    var sellPercentageIndex = headers.indexOf('SELL%');
    var sellIndex = headers.indexOf('Sell');
    var commissionIndex = headers.indexOf('Commission');

    var result = {
      categories: [],
      noOfGoods: [],
      mktPrice: [],
      soldPrice: [],
      estSoldPrice: [],
      sellPercentage: [],
      sell: [],
      commission: [],
    };

    for (var i = 1; i < data.length; i++) {
      if (data[i][empIdIndex] == empId) {
        result.categories.push(data[i][categoryIndex]);
        result.noOfGoods.push(data[i][noOfGoodsIndex]);
        result.mktPrice.push(data[i][mktPriceIndex]);
        result.soldPrice.push(data[i][soldPriceIndex]);
        result.estSoldPrice.push(data[i][estSoldPriceIndex]);
        result.sellPercentage.push(data[i][sellPercentageIndex]);
        result.sell.push(data[i][sellIndex]);
        result.commission.push(data[i][commissionIndex]);
      }
    }

    return result;
  } catch (error) {
    return { error: error.message };
  }
}

function getRightTableData(poolId, empId) {
  try {
    var sheet =
SpreadsheetApp.getActiveSpreadsheet().getSheetByName(poolId);
```

```
  if (!sheet) {
    throw new Error('Invalid Pool ID');
  }

  var dataRange = sheet.getDataRange();
  var data = dataRange.getValues();

  // Assuming the headers are in the first row
  var headers = data[0];
  var empIdIndex = headers.indexOf('EMP_ID');
  var categoryIndex = headers.indexOf('Category');
  var noOfGoodsIndex = headers.indexOf('NO_OF_GOODS');
  var mktPriceIndex = headers.indexOf('MKT_PRICE');
  var soldPriceIndex = headers.indexOf('SOLD_PRICE');
  var sellPercentageIndex = headers.indexOf('SELL%');
  var sellIndex = headers.indexOf('Sell');
  var commissionIndex = headers.indexOf('Commission');

  var result = {
    categories: [],
    noOfGoods: [],
    mktPrice: [],
    soldPrice: [],
    sellPercentage: [],
    sell: [],
    commission: [],
  };

  for (var i = 1; i < data.length; i++) {
    if (data[i][empIdIndex] == empId) {
      result.categories.push(data[i][categoryIndex]);
      result.noOfGoods.push(data[i][noOfGoodsIndex]);
      result.mktPrice.push(data[i][mktPriceIndex]);
      result.soldPrice.push(data[i][soldPriceIndex]);
      result.sellPercentage.push(data[i][sellPercentageIndex]);
      result.sell.push(data[i][sellIndex]);
      result.commission.push(data[i][commissionIndex]);
    }
  }
```

```
    return result;
  } catch (error) {
    return { error: error.message };
  }
}

function displayRightTable(result) {
  var rightTable =
document.getElementById('rightTable').getElementsByTagName('tbody')[0];

  // Clear previous table rows
  rightTable.innerHTML = '';

  // Define specified percentages and corresponding commissions
  var specifiedPercentages = [
    result.SELL1_Percentage,
    result.SELL2_Percentage,
    result.SELL3_Percentage,
    result.SELL4_Percentage,
    result.SELL5_Percentage
  ];
  var correspondingCommissions = [
    result.Sell1_Price,
    result.Sell2_Price,
    result.Sell3_Price,
    result.Sell4_Price,
    result.Sell5_Price
  ];

  // Add new rows for each category
  for (var i = 0; i < result.categories.length; i++) {
    var formattedSellPercentage = parseFloat(result.sellPercentage[i]) *
100;
    var commission = "NA";

    // Check if the "Sell" value corresponds to any specified percentage
    for (var j = 0; j < specifiedPercentages.length; j++) {
      if (formattedSellPercentage >= specifiedPercentages[j]) {
```

```
        commission = correspondingCommissions[j];
      }
    }

    var rightTableRow = '<tr>' +
      '<td>' + result.categories[i] + '</td>' +
      '<td>' + result.noOfGoods[i] + '</td>' +
      '<td>' + result.mktPrice[i] + '</td>' +
      '<td>' + result.soldPrice[i] + '</td>' +
      '<td>' + formattedSellPercentage.toFixed(2) + '%' + '</td>' +
      '<td>' + result.sell[i] + '</td>' +
      '<td>' + commission + '</td>' +
      '</tr>';
    rightTable.innerHTML += rightTableRow;
  }
}
```

Page.html

```
<!-- Page.html (HTML for the web app) -->
<!DOCTYPE html>
<html>

<head>
  <base target="_top">
  <style>
    /* Add your CSS styles for a visually appealing presentation */
  body {
```

```
      font-family: Arial, sans-serif;
      background-color: #f4f4f4;
      margin: 0;
      padding-top: 60px; /* Added top padding to account for the fixed
header */
      padding-left: 20px;
    }
    .header {
      position: fixed; /* Make header fixed */
      top: 0; /* Align header to the top */
      left: 0; /* Align header to the left */
      width: 100%; /* Header should span the full width of the page */
      background-color: #4CAF50;
      color: white;
      text-align: center;
      padding: 10px 0;
      font-size: 24px;
      box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);
      z-index: 1000; /* Ensure the header is above all other elements */
    }

.container {
  display: flex;
  flex-wrap: wrap; /* Allow items to wrap to the next line */
}

.basic-details,
.additional-details,
.right-table,
.editable-table {
  margin: 10px;
}

.basic-details,
.right-table {
  width: calc(30% - 20px); /* 30% width with margin */
}

/* Additional Details Container */
```

```css
.additional-details-container {
  width: calc(70% - 20px);
  margin: 10px;
}

/* Additional Details */
.additional-details {
  width: 60%;
  padding: 20px;
  background-color: #fff;
  border: 1px solid #ddd;
  border-radius: 4px;
}

/* Labels and input fields in Employee Details */
.additional-details label,
.additional-details select,
.additional-details input {
  box-sizing: border-box;
  width: calc(50% - 20px);
  padding: 10px;
  margin-bottom: 10px;
  border: 1px solid #ccc;
  border-radius: 4px;
}

/* 'Get Details' button in Additional Details */
.additional-details button {
  width: calc(100% - 20px);
  padding: 12px;
  background-color: #4CAF50;
  color: #fff;
  border: none;
  border-radius: 4px;
  cursor: pointer;
}
```

```
/* Styles for the employee details container, left-aligned and with some styling */
/* Styles for select dropdown and input field within the employee details container */
/* Styles for select dropdown and input field within the employee details container */
.employee-details select,
.employee-details input {
  box-sizing: border-box; /* Include padding and border in the element's total width */
  width: calc(100% - 0px); /* Set the width to 100% and adjust for padding */
  padding: 12px; /* Add padding to input fields for a comfortable feel */
  margin-bottom: 16px; /* Add spacing below input fields for better separation */
  border: 1px solid #ccc; /* Add a light border to input fields for visibility */
  border-radius: 4px; /* Add rounded corners to input fields */
}

/* Styles for the 'Get Details' button within the employee details container */
.employee-details button {
  width: calc(100% - 0px); /* Take up the full width of the container and adjust for padding */
  padding: 12px; /* Add padding to the button for a clickable area */
  background-color: #4CAF50; /* Green background color for a visually appealing button */
  color: #fff; /* White text color for better contrast */
  border: none; /* Remove default button border */
  border-radius: 4px; /* Add rounded corners to the button */
  cursor: pointer; /* Change cursor to a pointer on hover for better usability */
}

/* Styles for the employee details container */
.employee-details {
  width: 100px; /* Adjust the width as needed */
  margin: 10px; /* Add margin for better spacing */
```

```
  padding: 20px; /* Add padding for better visual appeal */
  background-color: #fff; /* White background for a clean look */
  border: 1px solid #ddd; /* Add a light border for visibility */
  border-radius: 4px; /* Add rounded corners for a modern look */
}

/* Styles for the employee details container */
.employee-details {
  width: 300px; /* Adjust the width as needed */
  margin: 10px; /* Add margin for better spacing */
  padding: 10px; /* Add padding for better visual appeal */
  background-color: #fff; /* White background for a clean look */
  border: 1px solid #ddd; /* Add a light border for visibility */
  border-radius: 4px; /* Add rounded corners for a modern look */
}

/* Styles for labels in the employee details container */
.employee-details label {
  font-weight: bold; /* Make labels bold for emphasis */
  margin-bottom: 5px; /* Add spacing below labels for separation */
  display: block; /* Display labels as blocks for better alignment */
}

/* Styles for non-editable values in the employee details container */
.employee-details .field-like-value {
  width: 300%; /* Take up the full width of the container */
  padding: 1px; /* Add padding for better visual appeal */
  background-color: #f4f4f4; /* Light gray background for non-editable
values */
  border: 1px solid #ddd; /* Add a light border for visibility */
  border-radius: 4px; /* Add rounded corners for a modern look */
  margin-bottom: 16px; /* Add spacing below values for better separation */
  display: inline-block; /* Display values as inline-block for a field-like
layout */
}

/* Hover effect for the 'Get Details' button */
.employee-details button:hover {
```

```
  background-color: #45a049; /* Darker green color on hover for a subtle
effect */
}



/* Styles for the additional-details cells in the first column */
.additional-details td:first-child {
  white-space: nowrap; /* Prevent text from wrapping */
}

/* Styles for the additional-details section */
.additional-details table {
  width: 70%; /* Make the table take up the full width */
  margin-top: 0px; /* Add spacing between tables */
  border-collapse: collapse; /* Collapse borders to avoid double borders */
  box-sizing: border-box; /* Include padding and border in the element's
total width */
}

/* Styles for the additional-details header (top row) */
.additional-details th {
  background-color: #4CAF50; /* Green background color for the header row
*/
  color: white; /* White text color for better contrast */
  padding: 12px;
  border: 1px solid #ddd;
  text-align: left;
}

/* Styles for the additional-details rows (excluding header) */
.additional-details tr:not(:first-child) {
  border: 1px solid #ddd;
}

/* Styles for the additional-details cells */
.additional-details td {
  padding: 5px;
  border: 1px solid #ddd;
```

```
  word-break: break-word;
  white-space: normal;
  color: black; /* Default black text color for cells */
}

/* Highlight even rows in the additional-details table */
.additional-details tr:nth-child(even) {
  background-color: #f9f9f9;
}

/* Hover effect for rows in the additional-details table */
.additional-details tr:hover {
  background-color: #e5e5e5;
}

/* Styles for values in the additional-details cells */
.additional-details td.numeric-value {
  text-align: right;
}

/* Round numeric values to 2 decimal points */
.additional-details td.numeric-value span {
  display: block;
  white-space: nowrap; /* Prevent text from wrapping */
  overflow: hidden;
  text-overflow: ellipsis;
  max-width: 100%;
  text-align: right;
  padding-right: 5px;
}
```

```css
/* Editable Table Container */
.editable-table-container {
  width: 60%;
  margin: 0;
}

/* Editable Table */
.editable-table {
  width: 91%;
  margin-top: 0;
  border-collapse: collapse;
  background-color: #fff;
}

/* Editable Table Header */
.editable-table th {
  background-color: #4CAF50;
  color: white;
  padding: 12px;
  text-align: left;
  border: 1px solid #ddd;
}

/* Editable Table Rows */
.editable-table tr {
  border: 1px solid #ddd;
}

/* Editable Table Cells */
.editable-table td {
  padding: 10px;
  border: 1px solid #ddd;
  word-break: break-word;
  white-space: nowrap;
  overflow: hidden;
  text-overflow: ellipsis;
}

/* Highlight the even rows in the Editable Table */
```

```css
.editable-table tr:nth-child(even) {
  background-color: #f9f9f9;
}

/* Hover effect for the Editable Table rows */
.editable-table tr:hover {
  background-color: #e5e5e5;
}

/* Styles for values in the Editable Table cells */
.editable-table td.numeric-value {
  text-align: right;
}

/* Add a bit of spacing between cells */
.editable-table td,
.editable-table th {
  box-sizing: border-box;
}

/* Responsive table styling */
@media (max-width: 600px) {
  .editable-table-container {
    width: 100%;
  }
}
```

```
  </style>
</head>

<body>
  <div class="header">Incentive Calculator</div>
  <div class="container">
    <div class="employee-details">
      <h3> Employee Details </h3>
      <label for="poolId">Select Your Pool:</label>
      <select id="poolId">
        <option value="ID1-1">ID1-1</option>
        <option value="ID2">ID2</option>
      </select>
      <br>
      <label for="empId">Enter EMP ID:</label>
      <input type="text" id="empId">
      <br>
      <button onclick="getDetails()">Get Details</button>
      <div id="result"></div>
    </div>

    <div class="additional-details">

      <h3>Right Table</h3>
      <table id="rightTable">
        <thead>
          <tr>
            <th>Category</th>
            <th>NO_OF_GOODS</th>
            <th>MKT_PRICE</th>
            <th>SOLD_PRICE</th>
            <th>SELL%</th>
            <th>Sell</th>
            <th>Commission</th>
          </tr>
        </thead>
        <tbody></tbody>
      </table>
    </div>
```

```
  </div>

  <div class="editable-table">
    <h3>Editable Table</h3>
    <table id="editableTable">
      <thead>
        <tr>
          <th>Category</th>
          <th>NO_OF_GOODS</th>
          <th>MKT_PRICE</th>
          <th>SOLD_PRICE</th>
          <th>EST_SOLD_PRICE</th>
          <th>SELL%</th>
          <th>Sell</th>
          <th>Commission</th>
        </tr>
      </thead>
      <tbody></tbody>
    </table>
  </div>

<script>
  function getDetails() {
    var poolId = document.getElementById('poolId').value;
    var empId = document.getElementById('empId').value;

    google.script.run.withSuccessHandler(displayResult).getData(poolId,
empId);

google.script.run.withSuccessHandler(displayEditableTable).getEditableTable
Data(poolId, empId);

google.script.run.withSuccessHandler(displayRightTable).getRightTableData(p
oolId, empId);
  }

  function displayResult(result) {
    var resultDiv = document.getElementById('result');
```

```
    if (result.error) {
      resultDiv.innerHTML = '<p>Error: ' + result.error + '</p>';
    } else {
      resultDiv.innerHTML = '<p>First Name: ' + result.firstName + '</p>' +
        '<p>User Role: ' + result.userRole + '</p>' +
        '<p>Total No of Goods: ' + result.noOfGoods + '</p>' +
        '<p>Sum of MKT_PRICE: ' + result.mktPriceSum + '</p>' +
        '<p>Sum of SOLD_PRICE: ' + result.soldPriceSum + '</p>';
    }
  }

  function displayEditableTable(result) {
    var editableTable =
document.getElementById('editableTable').getElementsByTagName('tbody')[0];

    // Clear previous table rows
    editableTable.innerHTML = '';

    // Add new rows for each category
    for (var i = 0; i < result.categories.length; i++) {
      var formattedSellPercentage = (parseFloat(result.sellPercentage[i]) *
100).toFixed(2) + '%';
      var editableTableRow = '<tr>' +
        '<td>' + result.categories[i] + '</td>' +
        '<td>' + result.noOfGoods[i] + '</td>' +
        '<td>' + result.mktPrice[i] + '</td>' +
        '<td>' + result.soldPrice[i] + '</td>' +
        '<td><input type="text" value="' + result.estSoldPrice[i] +
'"></td>' +
        '<td>' + formattedSellPercentage + '</td>' +
        '<td>' + result.sell[i] + '</td>' +
        '<td>' + result.commission[i] + '</td>' +
        '</tr>';
      editableTable.innerHTML += editableTableRow;
    }
  }

  function displayRightTable(result) {
```

```
    var rightTable =
document.getElementById('rightTable').getElementsByTagName('tbody')[0];

    // Clear previous table rows
    rightTable.innerHTML = '';

    // Add new rows for each category
    for (var i = 0; i < result.categories.length; i++) {
      var formattedSellPercentage = (parseFloat(result.sellPercentage[i]) *
100).toFixed(2) + '%';
      var rightTableRow = '<tr>' +
        '<td>' + result.categories[i] + '</td>' +
        '<td>' + result.noOfGoods[i] + '</td>' +
        '<td>' + result.mktPrice[i] + '</td>' +
        '<td>' + result.soldPrice[i] + '</td>' +
        '<td>' + formattedSellPercentage + '</td>' +
        '<td>' + result.sell[i] + '</td>' +
        '<td>' + result.commission[i] + '</td>' +
        '</tr>';
      rightTable.innerHTML += rightTableRow;
    }
  }
</script>




</body>

</html>
```