Step 1: Create a Machine Learning Model

In your Jupyter Notebook cell, you can create and save a model as follows:

```python
import pickle
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier

# Load the iris dataset (as an example)
data = load_iris()
X, y = data.data, data.target

# Split the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Create and train a simple Random Forest classifier (as an example)
model = RandomForestClassifier(n_estimators=100, random_state=42)
model.fit(X_train, y_train)

# Save the model to a file using Pickle
with open('model.pkl', 'wb') as model_file:
    pickle.dump(model, model_file)
```

This code will create and save a Random Forest classifier model as `model.pkl` in the same directory where your Jupyter Notebook is located.

Step 2: Create a Flask Web Application

Next, you need to create a Flask web application. Here's the directory structure you should have:

```
my_flask_app/
├── app.py
├── templates/
|   ├── index.html
├── model.pkl
```

In your Jupyter Notebook cell, you can create a simple Flask application as follows:

```python
from flask import Flask, render_template, request
import pickle

app = Flask(__name__)

# Load the Pickle model
with open('model.pkl', 'rb') as model_file:
    model = pickle.load(model_file)

@app.route('/')
def index():
    return render_template('index.html')

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

```
def predict():
    try:
        # Get input data from the HTML form
        feature1 = float(request.form.get('feature1'))
        feature2 = float(request.form.get('feature2'))

        # Make a prediction using the model
        prediction = model.predict([[feature1, feature2]])[0]

        # Pass the prediction to the HTML template
        return render_template('index.html', prediction=prediction)

    except Exception as e:
        return render_template('index.html', error_message=str(e))

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

Step 3: Create the HTML Template (templates/index.html)

In the `templates` directory, create an HTML template file named `index.html` as follows:

```
<!DOCTYPE html>
<html>
<head>
    <title>Machine Learning App</title>
    <style>
        body {
```

```
  font-family: Arial, sans-serif;
  margin: 20px;
  padding: 20px;
}

h1 {
  font-size: 24px;
  margin-bottom: 20px;
}

form {
  margin-bottom: 20px;
}

label {
  display: block;
  font-weight: bold;
  margin-bottom: 5px;
}

input[type="text"] {
  width: 100%;
  padding: 10px;
  margin-bottom: 10px;
  border: 1px solid #ccc;
  border-radius: 4px;
}

input[type="submit"] {
```

```
            background-color: #007bff;
            color: #fff;
            padding: 10px 20px;
            border: none;
            border-radius: 4px;
            cursor: pointer;
        }

        input[type="submit"]:hover {
            background-color: #0056b3;
        }

        h2 {
            font-size: 20px;
            margin-top: 20px;
        }

        p {
            font-size: 16px;
        }

        .error {
            color: red;
        }
    </style>
</head>
<body>
    <h1>Machine Learning App</h1>
    <form method="POST" action="/predict">
```

```
        <label for="feature1">Feature 1:</label>
        <input type="text" id="feature1" name="feature1" required>
        <br>
        <label for="feature2">Feature 2:</label>
        <input type="text" id="feature2" name="feature2" required>
        <br>
        <input type="submit" value="Predict">
    </form>
    {% if error_message %}
    <h2>Error:</h2>
    <p class="error">{{ error_message }}</p>
    {% endif %}
    {% if prediction %}
    <h2>Prediction:</h2>
    <p>{{ prediction }}</p>
    {% endif %}
</body>
</html>
```

Step 4: Run the Flask Application

Navigate to the `flask_pkl_app` directory in your terminal where the `app.py` file is located and run the Flask app:

```
python app.py
```

Now, your Flask application is running, and you can access it in your web browser at `http://localhost:5000`.

Understand Your Prediction Result:

```
if prediction == 0:
    prediction_label = "Setosa"
elif prediction == 1:
    prediction_label = "Versicolor"
elif prediction == 2:
    prediction_label = "Virginica"
else:
    prediction_label= "Unknown"
```

This setup includes exception handling in the Flask app, which will display any errors on the web page if there are any issues during the prediction process.