MAKALAH TUGAS BESAR

PENERAPAN ALGORITMA RANDOM FOREST CLASSIFIER UNTUK PREDIKSI STATUS STOK PENJUALAN PRODUK BOLEN CRISPY

Makalah ini ditulis untuk memenuhi persyaratan untuk ujian akhir semester

pada mata kuliah Machine Learning

Oleh:

Muhamad Azi Sudarya	1102221014
Muhamad Ridho Alfarizi	1101221116
Khaisar Rizki Maulana	1101221151
Nadia Chusnul Ikromah	1102221001
Ravina Indriyani	1102221003
Rosmawati	1102221023

TEKNIK INFORMATIKA

UNIVERSITAS BANTEN JAYA

2025

ABSTRAK

Permasalahan utama yang dihadapi UMKM seperti Bolen Crispy adalah pengelolaan stok yang kurang optimal, yang dapat mengakibatkan overstock atau understock. Penelitian ini bertujuan membangun sistem prediksi status stok menggunakan algoritma *Random Forest Classifier*, yang diklasifikasikan ke dalam tiga kategori: understock, normal, dan overstock. Metode yang digunakan meliputi pengumpulan data penjualan historis, preprocessing, rekayasa fitur, penyeimbangan data menggunakan SMOTE, pelatihan model, dan evaluasi performa melalui metrik klasifikasi. Hasil evaluasi menunjukkan bahwa model memiliki akurasi 93%, dengan *precision*, *recall*, dan *F1-score* tinggi, serta kesalahan klasifikasi yang rendah. Model mampu mengidentifikasi pola stok secara efektif dan dapat digunakan pelaku usaha untuk mengambil keputusan stok yang lebih akurat. Penelitian ini menyarankan agar sistem diperluas dengan data eksternal dan diuji langsung dalam kegiatan operasional.

Kata Kunci: Algoritma, Klasifikasi, Machine Learning, Prediksi, Random Forest, Stok Produk, UMKM

ABSTRACT

One of the main problems faced by MSMEs such as Bolen Crispy is suboptimal stock management, which can lead to overstock or understock conditions. This study aims to build a stock status prediction system using the Random Forest Classifier algorithm, classifying stock into three categories: understock, normal, and overstock. The method involves collecting historical sales data, preprocessing, feature engineering, data balancing using SMOTE, model training, and performance evaluation using classification metrics. The evaluation results show that the model achieves 93% accuracy, with high precision, recall, and F1-score, and minimal classification errors. The model effectively identifies stock patterns and can assist business owners in making more accurate inventory decisions. This study suggests expanding the dataset with external variables and testing the system directly in real operational settings.

Keywords: *Algorithm, Classification, Machine Learning, MSMEs, Prediction, Product Stock, Random Forest*

BAB I

PENDAHULUAN

1.1 Latar Belakang

Perkembangan teknologi informasi telah membawa perubahan signifikan dalam berbagai aspek kehidupan, termasuk dalam sektor bisnis dan perdagangan. Salah satu tantangan utama yang dihadapi oleh pelaku usaha, khususnya sektor Usaha Mikro, Kecil, dan Menengah (UMKM), adalah bagaimana mengelola stok secara efisien agar sesuai dengan kebutuhan pasar. Kesalahan dalam pengelolaan stok dapat menyebabkan kelebihan *(overstock)* atau kekurangan *(understock)* persediaan, yang berdampak pada meningkatnya biaya operasional, pemborosan sumber daya, serta penurunan kepuasan pelanggan.

Usaha Bolen Crispy di Desa Pekalongan merupakan salah satu contoh usaha makanan lokal yang menghadapi tantangan tersebut. Produk yang ditawarkan memiliki tingkat permintaan yang fluktuatif, sehingga menyulitkan pemilik usaha dalam menentukan jumlah stok yang optimal. Dalam praktiknya, pengelolaan persediaan sering kali masih dilakukan secara manual atau berbasis intuisi, tanpa dukungan sistem yang mampu memberikan prediksi secara akurat. Kondisi ini dapat menyebabkan kerugian, baik dalam bentuk produk tidak terjual maupun kehilangan potensi penjualan.

Untuk mengatasi permasalahan tersebut, diperlukan penerapan teknologi yang mampu memberikan prediksi status stok secara sistematis dan berbasis data. Salah satu pendekatan yang dapat digunakan adalah machine learning, khususnya algoritma Random Forest Classifier. Algoritma ini memiliki keunggulan dalam menangani data kompleks, mengklasifikasikan data ke dalam kategori tertentu, serta menghindari overfitting melalui penggunaan ensemble pohon keputusan.

Dalam penelitian ini, algoritma Random Forest Classifier diterapkan untuk melakukan klasifikasi status stok menjadi tiga kategori, yaitu *understock*, normal, dan *overstock*. Pendekatan ini diharapkan dapat membantu pelaku usaha dalam mengambil keputusan yang lebih tepat dalam pengelolaan stok produk, sehingga dapat mengurangi risiko kerugian akibat kesalahan prediksi. Dengan adanya sistem berbasis web yang terintegrasi dengan algoritma ini, diharapkan pelaku usaha dapat memperoleh informasi

prediktif yang membantu dalam pengambilan keputusan strategis, mengurangi risiko kerugian, dan meningkatkan daya saing usaha.

1.2 Rumusan Masalah

1. Bagaimana cara mengklasifikasikan status stok produk Bolen Crispy menjadi kategori *understock*, normal, dan *overstock* secara akurat?
2. Seberapa tinggi tingkat akurasi prediksi yang dihasilkan oleh algoritma Random Forest Classifier?
3. Bagaimana sistem prediksi status stok berbasis web dapat membantu pelaku usaha dalam pengambilan keputusan pengelolaan persediaan?

1.3 Tujuan Penelitian

1. Membangun model klasifikasi status stok produk Bolen Crispy menggunakan algoritma Random Forest Classifier.
2. Menilai akurasi dan performa algoritma Random Forest Classifier dalam sistem stok penjualan.
3. Mengembangkan sistem prediksi status stok berbasis web yang dapat membantu pelaku usaha dalam mengelola persediaan secara lebih efisien dan strategis.

1.4 Manfaat Penelitian

Penelitian ini diharapkan dapat memberikan manfaat praktis bagi pelaku UMKM, khususnya dalam pengelolaan stok dan perencanaan produksi. Selain itu, secara akademik, penelitian ini dapat memperkaya literatur terkait penerapan machine learning di bidang manajemen bisnis lokal.

BAB II

TINJAUAN PUSTAKA

2.1 Machine Learning dan Random Forest Classifier

Machine learning adalah salah satu cabang dari kecerdasan buatan (*Artificial Intelligence*) yang berfokus pada pengembangan sistem atau algoritma yang mampu belajar dari data tanpa perlu diprogram secara eksplisit. Melalui proses pembelajaran dari pola-pola dalam data historis, machine learning memungkinkan sistem untuk membuat prediksi, klasifikasi, atau pengambilan keputusan secara otomatis berdasarkan data input baru.

Salah satu algoritma yang banyak digunakan dalam tugas klasifikasi adalah Random Forest Classifier. Random Forest termasuk dalam metode *ensemble learning*, yang menggabungkan sejumlah algoritma pembelajaran (dalam hal ini, pohon keputusan) untuk memperoleh hasil prediksi yang lebih akurat dan andal. Random Forest bekerja dengan membangun banyak pohon keputusan secara acak selama proses pelatihan, lalu menggabungkan hasil klasifikasi dari setiap pohon untuk menentukan output akhir berdasarkan prinsip voting mayoritas (*majority voting*).

Keunggulan dari Random Forest Classifier antara lain adalah kemampuannya dalam menangani dataset berukuran besar dan kompleks, ketahanannya terhadap overfitting, serta kestabilan performa meskipun terdapat data yang tidak seimbang atau mengandung noise. Karena sifatnya yang robust, algoritma ini sangat cocok digunakan untuk permasalahan klasifikasi seperti dalam prediksi status stok produk yang diklasifikasikan ke dalam kategori tertentu, seperti *understock*, normal, dan *overstock*.

2.2 Klasifikasi Stok Penjualan

Klasifikasi stok penjualan merupakan proses untuk mengelompokkan data penjualan berdasarkan kategori tertentu guna membantu dalam pengambilan keputusan manajemen persediaan. Dengan pendekatan ini, sistem dapat mengenali pola penjualan, seperti periode dengan permintaan tinggi *(peak season)* atau rendah *(low season)*, serta mendeteksi jenis produk dengan rotasi stok cepat atau lambat. Informasi ini penting untuk menghindari risiko *overstocking* (kelebihan stok) maupun *understocking* (kekurangan stok), yang dapat berdampak pada efisiensi operasional dan tingkat kepuasan pelanggan. Dengan menerapkan klasifikasi stok penjualan berbasis machine learning, pelaku usaha dapat meningkatkan akurasi prediksi dan pengendalian logistik secara real-time.

2.3 Penelitian Terkait

Beberapa penelitian terdahulu menunjukkan efektivitas Random Forest dalam prediksi penjualan. Misalnya, penelitian oleh Wang dan Aviles (2023) menunjukkan akurasi hingga 90% dalam prediksi bisnis. Sementara itu, Praveen et al. (2022) juga menyatakan bahwa Random Forest memiliki performa lebih baik dibanding regresi linier dan time series dalam konteks pengelolaan stok.

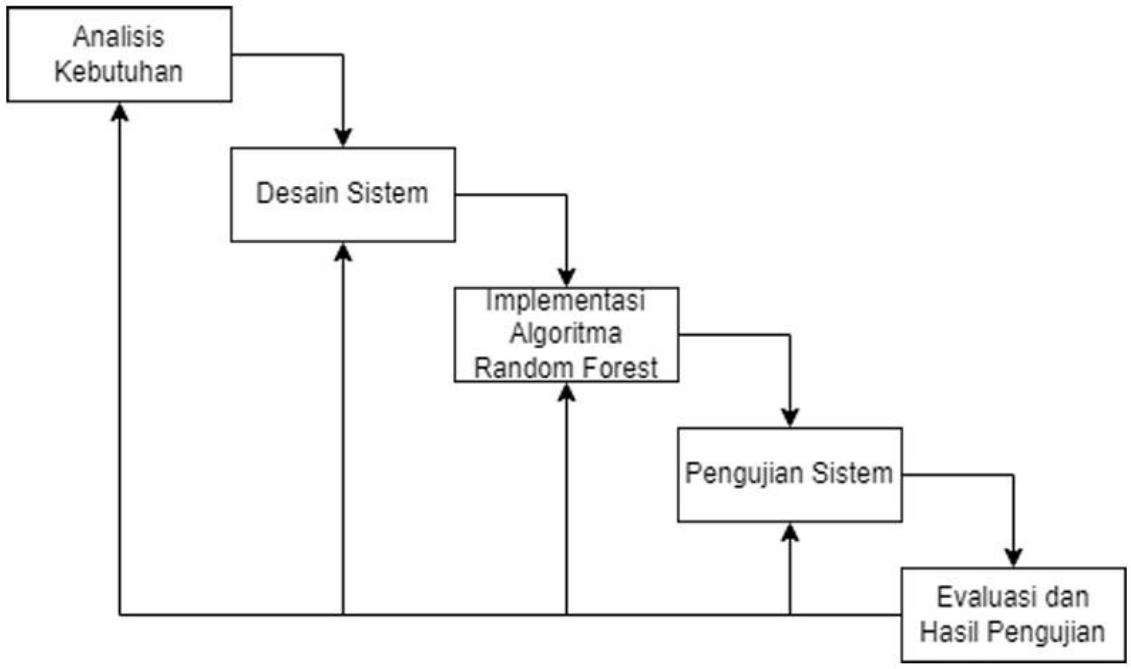
BAB III

METODOLOGI PENELITIAN

3.1 Metode Penelitian

Penelitian ini menggunakan pendekatan rekayasa perangkat lunak dengan menerapkan metode pengembangan sistem Waterfall. Metode ini dipilih karena memiliki tahapan yang sistematis, terstruktur, dan cocok digunakan untuk proses pengembangan sistem yang jelas dan tidak berubah secara signifikan selama proses berlangsung.

Waterfall merupakan metode pengembangan perangkat lunak klasik yang bersifat linier dan berurutan, di mana setiap tahap harus diselesaikan terlebih dahulu sebelum melanjutkan ke tahap berikutnya. Tahapan-tahapan dalam metode ini adalah sebagai berikut:



3.2 Tahapan Penelitian

3.2.1 Analisis Kebutuhan

Tahap ini bertujuan untuk mengidentifikasi kebutuhan sistem secara menyeluruh, baik kebutuhan fungsional maupun non-fungsional. Dalam konteks penelitian ini, kebutuhan yang dianalisis mencakup data penjualan historis dari tahun

2021 sampai 2024, nama produk, harga satuan, jumlah terjual, stok produk, serta kebutuhan pengguna terhadap sistem prediksi status stok penjualan .

3.2.2 Desain Sistem

Setelah kebutuhan sistem teridentifikasi dengan jelas, tahap selanjutnya adalah perancangan sistem. Perancangan ini meliputi desain alur sistem, struktur database, antarmuka pengguna. Dalam desain ini, kami juga merancang bagaimana algoritma Random Forest akan diimplementasikan dalam sistem, termasuk cara integrasi model Machine Learning ke dalam aplikasi web

3.2.3 Implementasi Algoritma Random Forest Classifier

Pada tahap ini dilakukan pengembangan dan penerapan algoritma Random Forest Classifier. Model dilatih menggunakan data historis yang telah diproses, dan digunakan untuk mengklasifikasikan status stok ke dalam kategori *understock*, normal, dan *overstock*.

3.2.4 Pengujian Sistem

Pengujian dilakukan untuk memastikan bahwa sistem berfungsi sesuai dengan yang diharapkan. Dalam tahap ini, saya menggunakan data pengujian untuk menguji kinerja algoritma Random Forest Classifier dalam memprediksi stok penjualan. Pengujian meliputi pengujian unit untuk setiap komponen *(unit testing)*, pengujian integrasi (*integration testing*) untuk memastikan bahwa modul bekerja dengan baik dalam keseluruhan sistem. Performa model Random Forest juga dievaluasi menggunakan metrik klasifikasi seperti akurasi, precision, recall, dan F1-score.

3.2.5 Evaluasi dan Pemeliharaan

Setelah sistem diimplementasikan, saya akan melakukan evaluasi untuk memastikan bahwa sistem berjalan sesuai dengan yang diharapkan dalam lingkungan operasional. Evaluasi ini juga mencakup pemantauan kinerja algoritma Random Forest secara berkala, serta melakukan perbaikan atau penyesuaian jika ditemukan masalah atau jika ada kebutuhan untuk mengadaptasi model dengan data baru yang muncul. Dengan mengikuti metode waterfall, penelitian ini dapat berjalan secara terstruktur dan memastikan bahwa setiap tahap telah dievaluasi dan divalidasi sebelum melanjutkan ke tahap berikutnya. Integrasi algoritma Random Forest dalam proses pengujian memastikan bahwa sistem prediksi stok penjualan produk Bolen Crispy berbasis web ini

tidak hanya berfungsi secara teknis, tetapi juga memberikan hasil yang akurat dan dapat diandalkan.

BAB IV

HASIL DAN PEMBAHASAN

4.1 Konversi Tipe Data

4.1.1 Tanggal

```
# Mengubah tipe data kolom tanggal menjadi datetime
df['Tanggal'] = pd.to_datetime(df['Tanggal'], dayfirst=True)

# Mengubah tipe data kolom tanggal menjadi integer
df['Tanggal Int'] = df['Tanggal'].astype('int64') // 10**6
```

Kolom tanggal diubah ke format datetime agar data bisa dianalisis berdasarkan waktu seperti melihat trend bulanan atau mingguan. Lalu dikonversi ke bentuk numerik (integer) agar dapat diproses oleh algoritma machine learning yang tidak bisa membaca format tanggal secara langsung.

4.1.2 Hari

```
# Konversi nama hari jadi angka
mapping_hari = {
    'Senin': 1,
    'Selasa': 2,
    'Rabu': 3,
    'Kamis': 4,
    'Jumat': 5,
    'Sabtu': 6,
    'Minggu': 7
}

df['Hari'] = df['Hari'].map(mapping_hari)

print(df[['Hari']].head(7))
```

```
   Hari
0     7
1     1
2     2
3     3
4     4
5     5
6     6
```

Nama hari dikonversi menjadi angka untuk mempermudah proses analisis dan pemodelan karena model machine learning dan analisis statistik tidak bisa memproses data kategorikal bertipe string secara langsung. Dengan mengubah nama hari menjadi nilai numerik, kita dapat melakukan analisis untuk mengenali pola dari fitur tersebut.

4.2 Feature Engineering

4.2.1 Status Stok

```
[ ]  # Membuat kolom status stok berdasarkan selisih antara stok produk dan jumlah terjual
     def get_status_stok(row):
         selisih = row['Stok Produk'] - row['Jumlah Terjual']
         if selisih <= 0:
             return 'Understock'
         elif selisih > 50:
             return 'Overstock'
         else:
             return 'Normal'

     df['Status Stok'] = df.apply(get_status_stok, axis=1)

     print(df[['Nama Produk', 'Jumlah Terjual', 'Stok Produk', 'Status Stok']].head())
```

```
          Nama Produk  Jumlah Terjual  Stok Produk Status Stok
0        Bolen Coklat              38          130   Overstock
1        Bolen Banana              80          125      Normal
2      Bolen Proltape              45           89      Normal
3   Bolen Coklat Keju             149          150      Normal
4  Bolen Cokju (Mini)             118          118  Understock
```

Fitur Status Stok dibuat berdasarkan selisih antara Stok Produk dan Jumlah Terjual. Tujuannya adalah untuk mengkategorikan kondisi stok menjadi tiga kelas:

- Understock: jika stok kurang atau habis (≤ 0),
- Normal: jika stok masih dalam batas aman (1 hingga 50),
- Overstock: jika stok berlebih (> 50).

Fitur ini penting untuk membantu dalam analisis ketersediaan produk, mengidentifikasi potensi kekurangan dan kelebihan stok dan fitur ini juga dapat dijadikan target variabel untuk model klasifikasi machine learning seperti Random Forest Classifier.

```
# Menampilkan proporsi tiap kategori status stok dalam bentuk persentase
print(df['Status Stok'].value_counts(normalize=True))
```

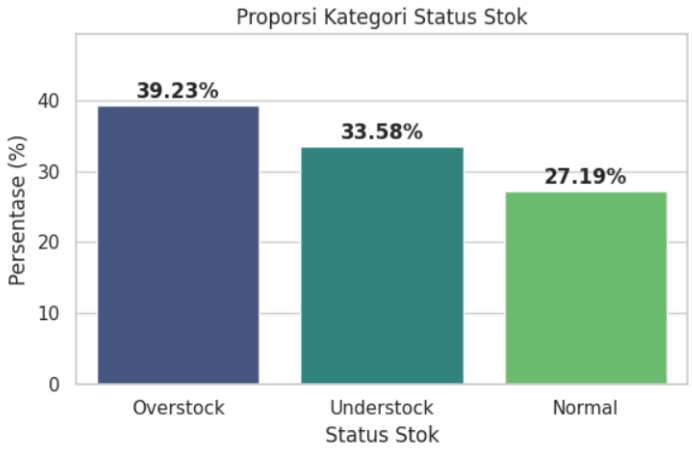
```
Status Stok
Overstock     0.392336
Understock    0.335766
Normal        0.271898
Name: proportion, dtype: float64
```

```
# Menghitung frekuensi tiap kategori status stok
status_counts = df['Status Stok'].value_counts(normalize=True)
status_percent = status_counts * 100

plt.figure(figsize=(6, 4))
ax = sns.barplot(x=status_percent.index, y=status_percent.values, palette='viridis')

for i, v in enumerate(status_percent.values):
    plt.text(i, v + 1, f"{v:.2f}%", ha='center', fontweight='bold')

plt.title('Proporsi Kategori Status Stok')
plt.xlabel('Status Stok')
plt.ylabel('Persentase (%)')
plt.ylim(0, max(status_percent.values) + 10)
plt.tight_layout()
plt.show()
```



Berdasarkan visualisasi di atas, proporsi kategori Status Stok menunjukkan bahwa sebagian besar data berada pada kondisi Overstock (39.23%), diikuti oleh Understock (33.58%), dan hanya 27.19% yang berada pada kondisi stok Normal. Hal ini mengindikasikan bahwa mayoritas stok belum dikelola secara optimal, dengan kecenderungan terjadi kelebihan atau kekurangan stok yang dapat berdampak pada efisiensi operasional dan potensi kehilangan penjualan.

4.3 Label Encoding

4.3.1 Status Stok

```
[ ] from sklearn.preprocessing import LabelEncoder

    # Mengubah fitur Status Stok menjadi nilai numerik menggunakan LabelEncoder
    le_status = LabelEncoder()
    df['Status Stok Encoded'] = le_status.fit_transform(df['Status Stok'])
    print(df[['Status Stok', 'Status Stok Encoded']].drop_duplicates())
```

```
  Status Stok  Status Stok Encoded
0   Overstock                   1
1      Normal                   0
4  Understock                   2
```

Selanjutnya, mengubah nilai kategorikal pada kolom Status Stok menjadi bentuk numerik menggunakan Label Encoder, menghasilkan kolom baru bernama Status Stok Encoded. Tujuan dari proses ini adalah agar data dapat diproses oleh algoritma machine learning yang umumnya tidak menerima input bertipe string atau kategorikal. Encoding ini memungkinkan model memahami nilai kategori dalam format yang dapat dihitung dan dianalisis.

4.3.2 Nama Produk

```
# Mengubah fitur Nama Produk menjadi nilai numerik menggunakan LabelEncoder
le = LabelEncoder()
df['Nama Produk Encoded'] = le.fit_transform(df['Nama Produk'])
print(df[['Nama Produk', 'Nama Produk Encoded']].drop_duplicates())
```

```
            Nama Produk  Nama Produk Encoded
0         Bolen Coklat                    2
1         Bolen Banana                    0
2       Bolen Proltape                    6
3    Bolen Coklat Keju                    3
4   Bolen Cokju (Mini)                    1
5    Bolen Keju (Mini)                    4
6  Bolen Pisang Coklat                    5
```

Kolom Nama Produk di encoding dengan tujuan untuk mempermudah proses analisis dan pemodelan saat pemilihan fitur dan target variabel. Dengan mengubah nama produk ke bentuk numerik, model bisa memahami dan memproses fitur tersebut dalam proses pelatihan dan prediksi.

4.4 Data Split

```
[ ] # Split data menjadi 80% data training dan 20% data testing
    fitur_rfc = df[['Tanggal Int', 'Hari', 'Nama Produk Encoded', 'Harga Satuan', 'Stok Produk', 'Jumlah Terjual']]
    target_rfc = df['Status Stok Encoded']

    X_train_rfc, X_test_rfc, y_train_rfc, y_test_rfc = train_test_split(
        fitur_rfc,
        target_rfc,
        test_size=0.2,
        random_state=42,
        stratify=target_rfc
    )
```

Data dibagi menjadi dua bagian: 80% untuk training dan 20% untuk testing. Proses ini dilakukan untuk memisahkan data yang akan digunakan untuk melatih model dan mengevaluasi performa model.

Teknik stratified splitting digunakan agar distribusi kelas pada target Status Stok Encoded tetap seimbang di kedua subset. Ini penting untuk mencegah model menjadi bias terhadap kelas tertentu, terutama ketika dataset memiliki ketidakseimbangan antar kelas.

4.5 SMOTE

```
from imblearn.over_sampling import SMOTE

# Menyeimbangkan data target dengan SMOTE dan cek distribusinya
smote = SMOTE(random_state=42)
X_train_smote, y_train_smote = smote.fit_resample(X_train_rfc, y_train_rfc)

print("Distribusi sebelum SMOTE:\n", pd.Series(y_train_rfc).value_counts())
print("\nDistribusi setelah SMOTE:\n", pd.Series(y_train_smote).value_counts())
```

Sebelum diterapkannya SMOTE, distribusi kelas pada target Status Stok Encoded terlihat tidak seimbang, di mana kelas 1 (Overstock) mendominasi dengan 344 data, kelas 2 (Understock) memiliki 294 data, sementara kelas 0 (Normal) hanya memiliki 238 data. Ketidakseimbangan ini dapat menyebabkan model cenderung bias dan kurang akurat dalam memprediksi kelas minoritas.

Setelah dilakukan SMOTE (Synthetic Minority Over-sampling Technique), jumlah data untuk setiap kelas disamakan menjadi 344, menghasilkan dataset yang seimbang. Dengan data yang seimbang, model klasifikasi memiliki peluang yang lebih adil dalam mempelajari pola dari setiap kelas dan menghasilkan prediksi yang lebih akurat dan tidak bias terhadap kelas mayoritas.

4.6 Data Modelling

```
# Melatih model dan melakukan prediksi pada data uji
model_rfc = RandomForestClassifier(random_state=42, class_weight='balanced')
model_rfc.fit(X_train_smote, y_train_smote)

y_pred_rfc = model_rfc.predict(X_test_rfc)
```

Model Random Forest dilatih menggunakan data pelatihan yang telah diseimbangkan dengan teknik SMOTE. Setelah proses pelatihan, model digunakan untuk memprediksi data uji. Langkah ini merupakan tahap inti untuk menguji seberapa baik model mampu mengenali pola dari data dan memprediksi kategori status stok produk (understock, normal, overstock) secara akurat.

4.7 Data Evaluation

```python
# Menampilkan metrik evaluasi model
accuracy = accuracy_score(y_test_rfc, y_pred_rfc)
precision = precision_score(y_test_rfc, y_pred_rfc, average='weighted')
recall = recall_score(y_test_rfc, y_pred_rfc, average='macro')
f1 = f1_score(y_test_rfc, y_pred_rfc, average='macro')

# Output hasil evaluasi
print("Accuracy :", round(accuracy, 2))
print("Precision:", round(precision, 2))
print("Recall   :", round(recall, 2))
print("F1 Score :", round(f1, 2))
```

```
Accuracy : 0.93
Precision: 0.93
Recall   : 0.92
F1 Score : 0.93
```

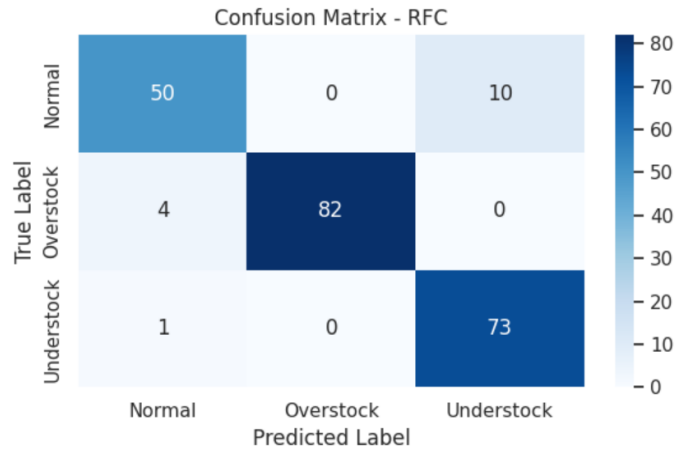
Model klasifikasi yang dibangun menunjukkan performa yang bagus dengan akurasi, precision, dan F1 score sebesar 93%, serta recall sebesar 92%. Hasil ini mengindikasikan bahwa model mampu mengklasifikasikan status stok produk secara konsisten, baik dalam mengenali maupun memprediksi kategori dengan akurasi tinggi.

4.8 Confusion Matrix

```python
# Menampilkan confusion matrix untuk mengevaluasi model Random Forest Classifier
cm = confusion_matrix(y_test_rfc, y_pred_rfc)

label_kelas = ['Normal', 'Overstock', 'Understock']

plt.figure(figsize=(6, 4))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', xticklabels=label_kelas, yticklabels=label_kelas)
plt.title('Confusion Matrix - RFC')
plt.xlabel('Predicted Label')
plt.ylabel('True Label')
plt.tight_layout()
plt.show()
```



Model mampu mengklasifikasikan sebagian besar data dengan benar, ditunjukkan oleh tingginya nilai diagonal (benar prediksi):

- 50 data kategori Normal diprediksi dengan benar.
- 82 data kategori Overstock diprediksi dengan benar.

- 73 data kategori Understock diprediksi dengan benar.

Kesalahan klasifikasi terjadi sebagian kecil:

- 10 data kategori Normal diprediksi sebagai Understock.
- 4 data Overstock diprediksi sebagai Normal.
- 1 data Understock diprediksi sebagai Normal.

Secara keseluruhan, confusion matrix ini menunjukkan bahwa model Random Forest Classifier memiliki performa klasifikasi yang sangat baik, dengan jumlah prediksi yang salah relatif kecil dibandingkan total data.

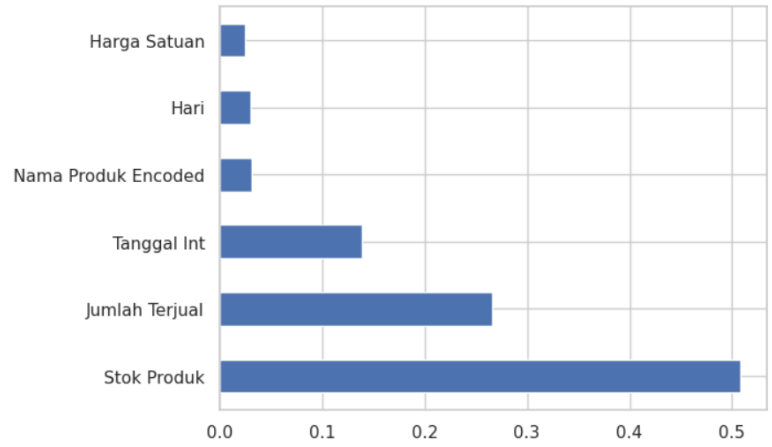
4.9 Classification Report

```
# Menampilkan Classification Report dari model RFC
print(classification_report(y_test_rfc, y_pred_rfc))
```

```
              precision    recall  f1-score   support

           0       0.91      0.83      0.87        60
           1       1.00      0.95      0.98        86
           2       0.88      0.99      0.93        74

    accuracy                           0.93       220
   macro avg       0.93      0.92      0.93       220
weighted avg       0.93      0.93      0.93       220
```



Berdasarkan grafik feature importance dari model Random Forest Classifier, fitur yang paling berpengaruh terhadap hasil prediksi adalah Stok Produk dan Jumlah Terjual, yang menunjukkan bahwa kondisi awal stok dan angka penjualan sangat menentukan status stok akhir (understock, normal, atau overstock). Sementara itu, fitur seperti Tanggal Int memiliki pengaruh sedang, menunjukkan adanya pola waktu tertentu. Fitur lain seperti Nama Produk

Encoded, Hari, dan Harga Satuan memiliki kontribusi rendah, sehingga perannya terhadap prediksi stok relatif kecil.

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

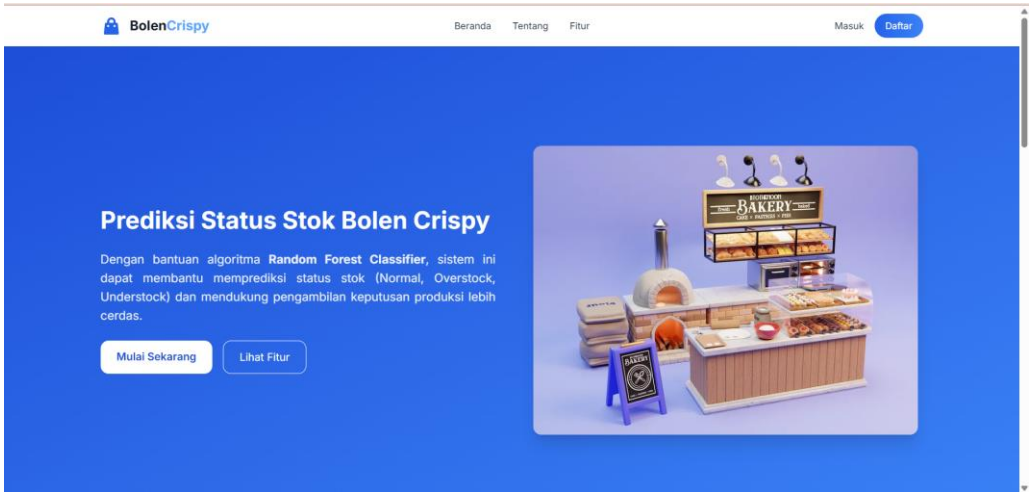
Penerapan algoritma Random Forest Classifier berhasil memberikan hasil prediksi yang akurat untuk klasifikasi status stok produk Bolen Crispy ke dalam kategori *understock*, normal, dan *overstock*. Dengan akurasi mencapai 93%, model ini diharapkan mampu membantu pelaku UMKM dalam mengelola stok secara lebih efisien dan mengurangi risiko kerugian akibat kelebihan atau kekurangan stok. Sistem yang dibangun juga telah dilengkapi dengan visualisasi dan evaluasi kinerja model, serta dapat digunakan kembali untuk prediksi di masa depan.

5.2 Saran

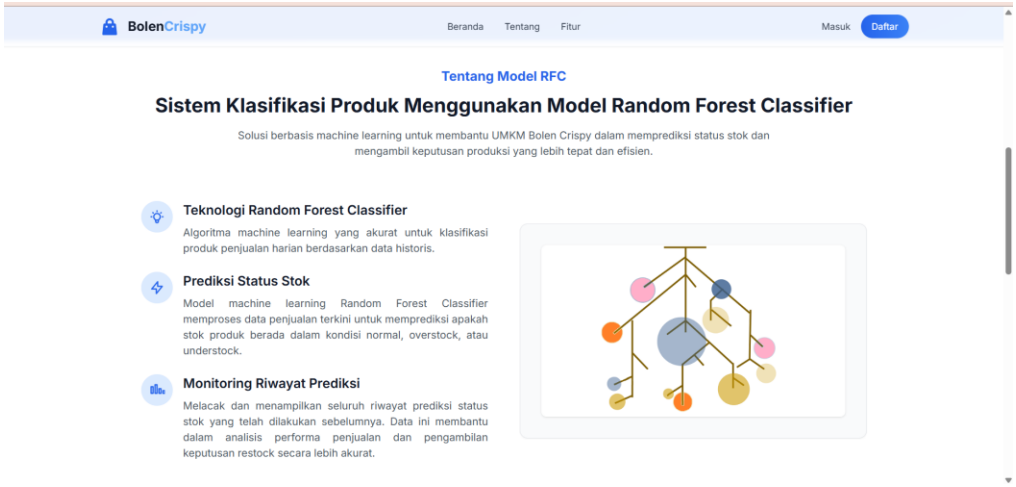
Agar sistem prediksi status stok semakin optimal, disarankan untuk memperluas data historis yang digunakan, menambahkan variabel eksternal seperti promosi atau hari libur yang dapat mempengaruhi penjualan, serta melakukan evaluasi dan pelatihan ulang model secara berkala agar tetap relevan dengan kondisi terkini. Selain itu, sistem juga sebaiknya diuji secara langsung dalam operasional usaha untuk memastikan manfaatnya dalam pengambilan keputusan stok penjualan harian secara nyata.

LAMPIRAN

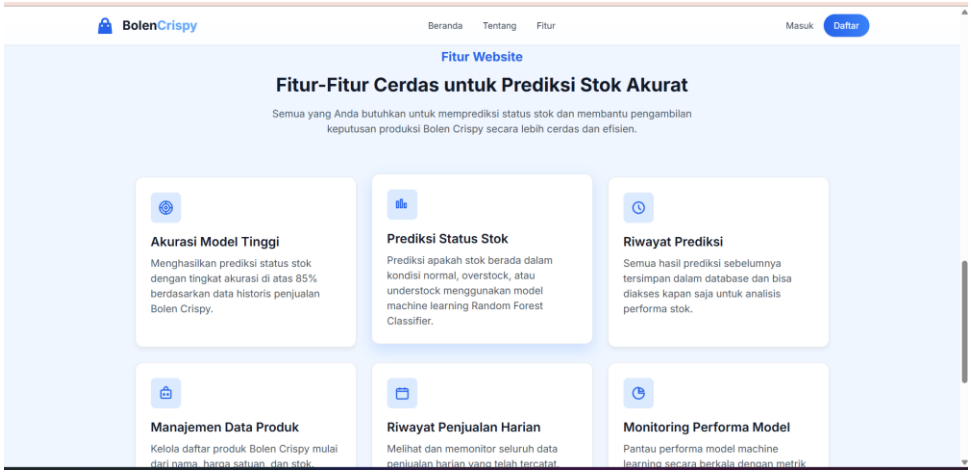
Halaman Utama Bolen Crispy



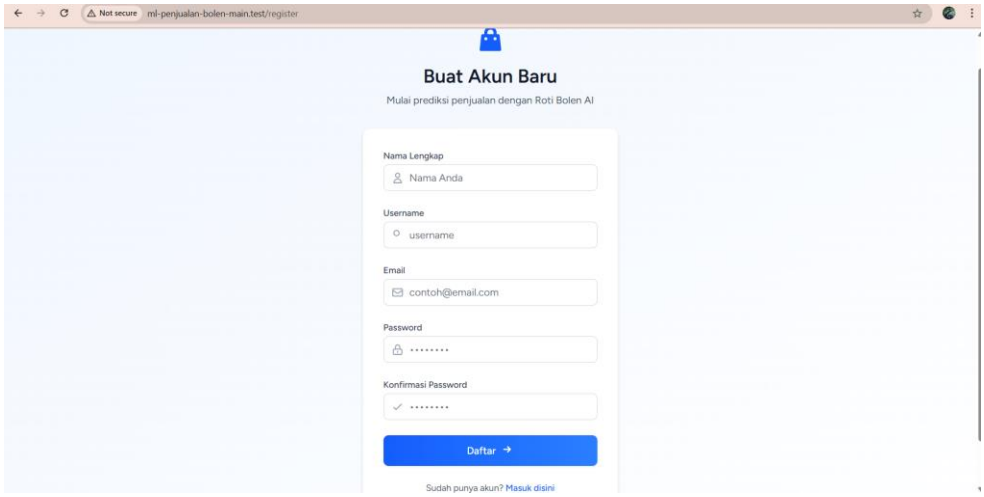
Halaman Tentang Model RFC Pada Sistem Bolen Crispy



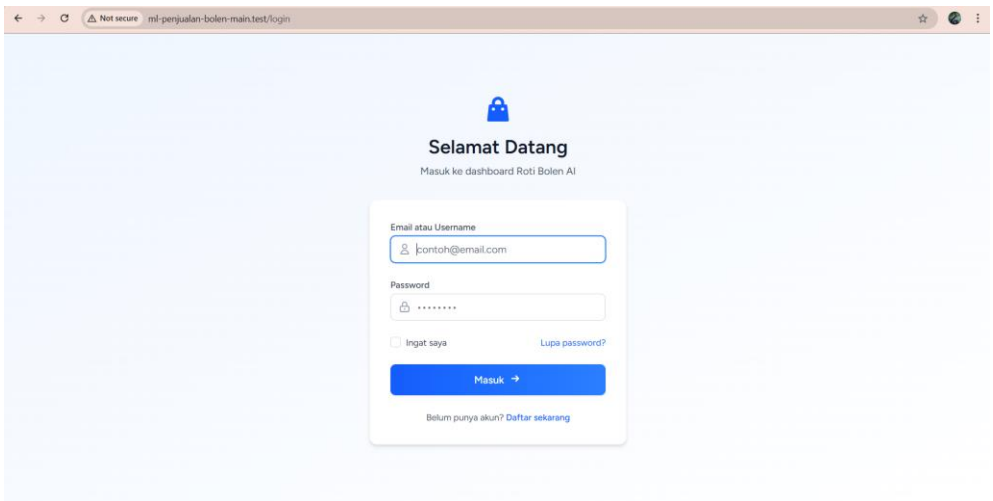
Halaman Fitur Website Bolen Crispy



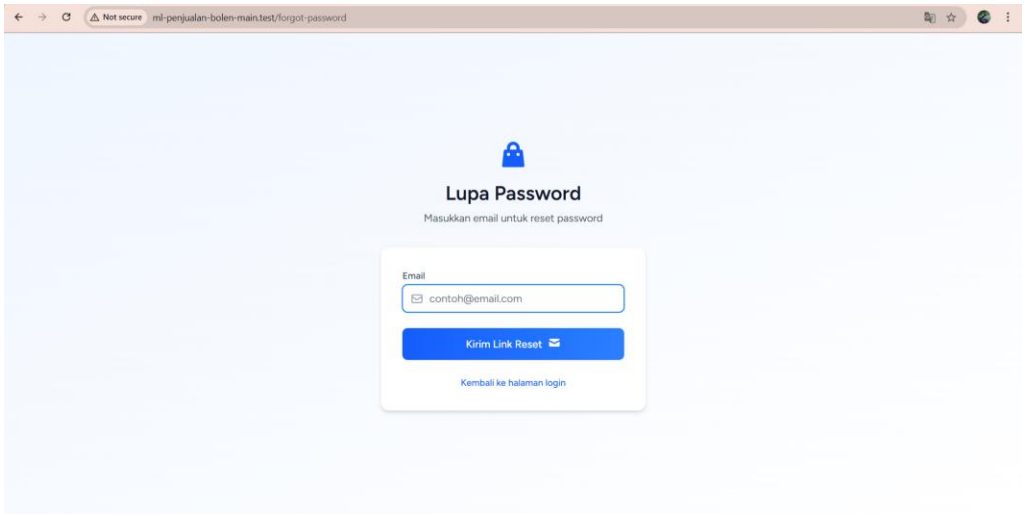
Halaman Registrasi



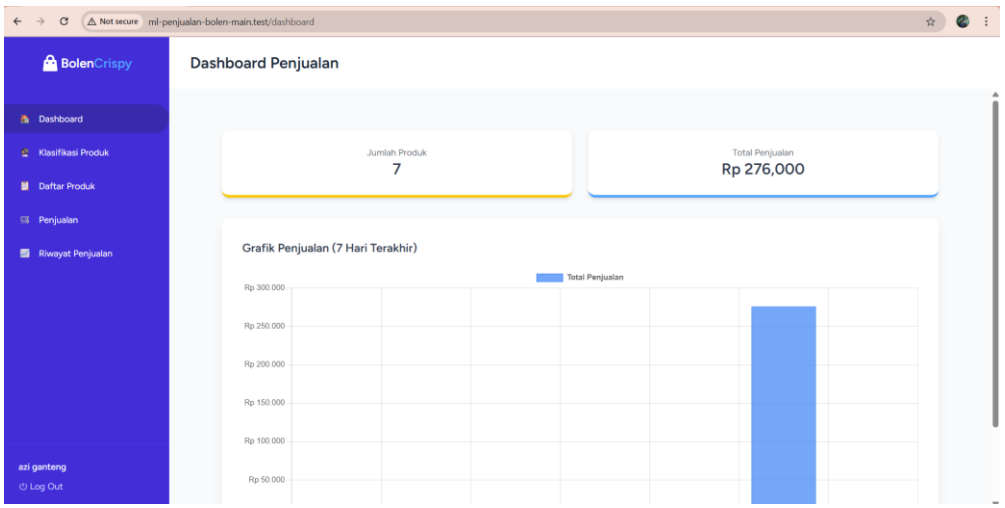
Halaman Login



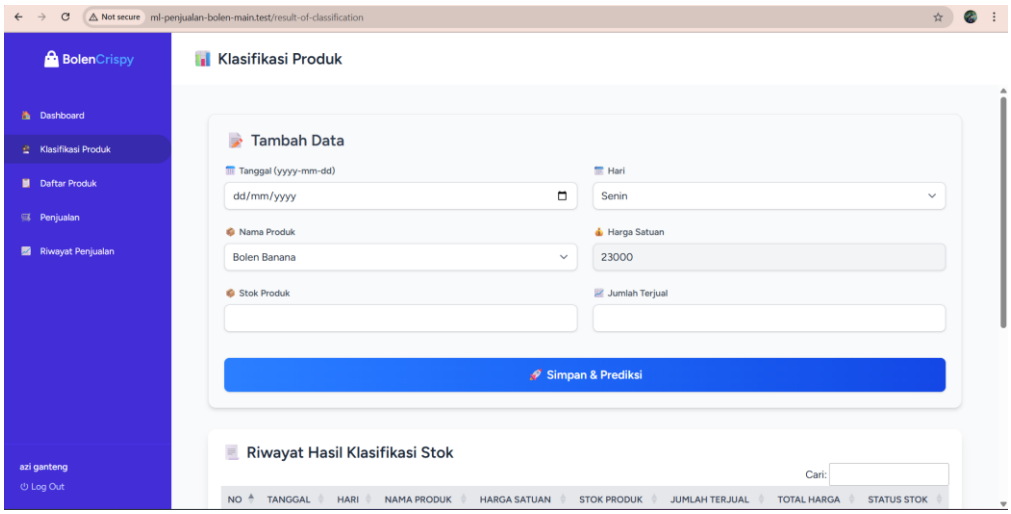
Halaman Lupa Password



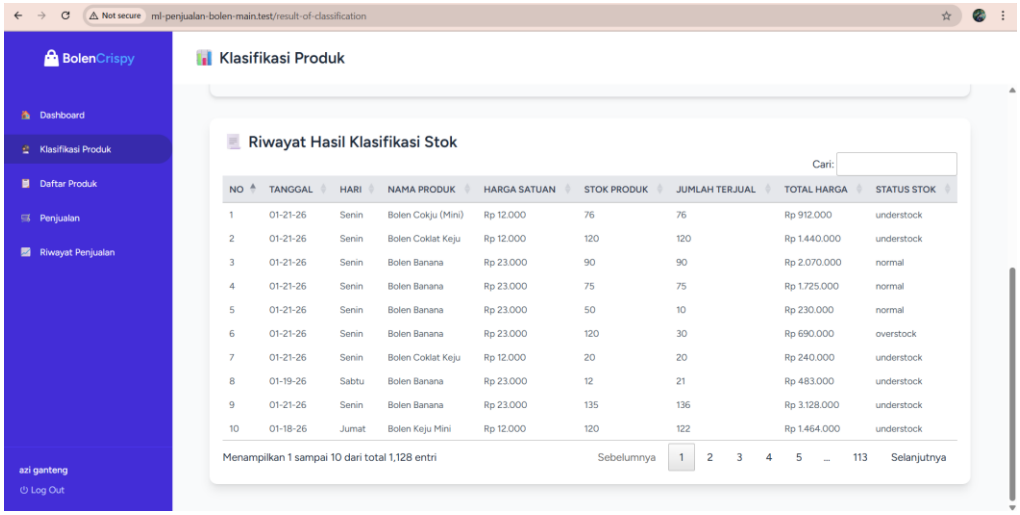
Halaman Dasboard



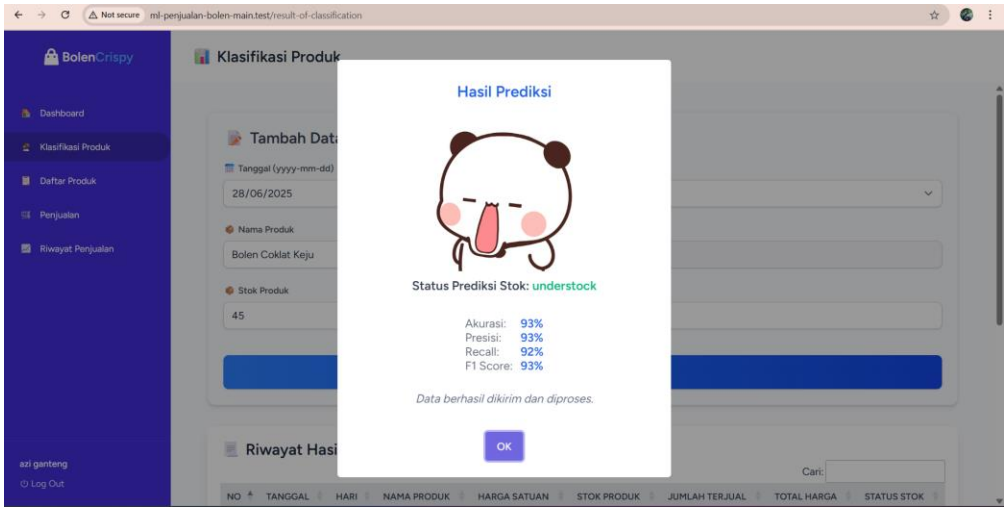
Halaman Input Untuk Klasifikasi Produk Berdasarkan Status Stok (Main Feature)



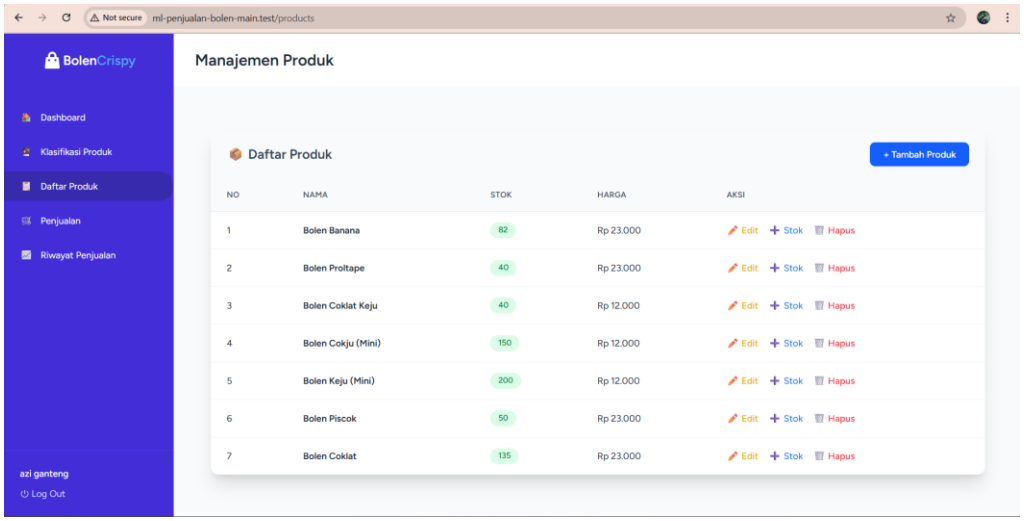
Riwayat Hasil Klasifikasi Produk (Main Feature)



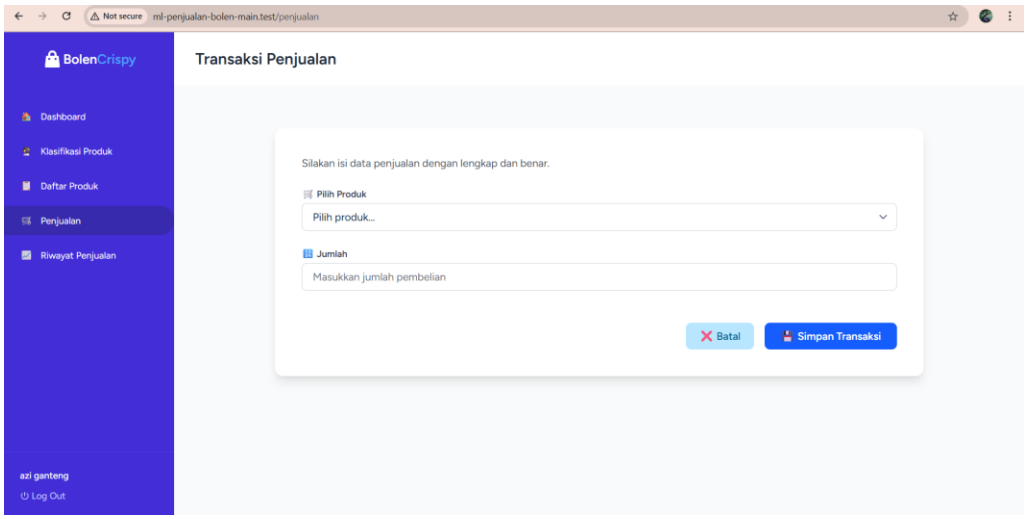
Hasil Klasifikasi Fitur (Main Feature)



Halaman Manajemen Produk



Halaman Transaksi Penjualan



Halaman Riwayat Penjualan