

TechNest Business Report

Presented by:

Team Member	Team Member Tasks
Nithin Chakravarthi	Product Report
Saif	Business Report
Pavan Bhargav Gorli	Product Report
Yasaswini Sri	Business Report

1. Problem Statement

The retail industry, particularly in the electronics sector, faces significant challenges in bridging the gap between local retailers and online consumers. Traditional retailers struggle to compete with e-commerce giants due to limited digital presence, price transparency issues, and difficulty in attracting tech-savvy customers. Consumers, on the other hand, often lack access to local deals, accurate price comparisons, and personalized product recommendations.

This disconnect results in:

- **Reduced Sales for Local Retailers:** Retailers find it challenging to drive foot traffic and convert online searches into purchases.
- **Lack of Competitive Pricing Insights:** Customers are unaware of local store prices and rely on large online platforms.
- **Inefficient Product Discovery:** Consumers struggle to find the best deals from nearby retailers, leading to a loss of potential savings.

To address these challenges, TechNest aims to provide an AI-driven platform that enhances price transparency, facilitates digital transformation for retailers, and optimizes the shopping experience for consumers.

2. Present Market Overview

The retail industry is undergoing a significant transformation due to the increasing adoption of digital solutions and e-commerce dominance. While large online retailers offer competitive pricing and convenience, local brick-and-mortar retailers struggle to attract customers due to limited online presence and lack of price transparency.

2.1 Key Market Challenges

- **Price Competition:** E-commerce giants offer lower prices due to economies of scale, making it hard for local retailers to compete.
- **Lack of Digital Infrastructure:** Many small retailers lack online visibility, limiting their potential customer base.

- **Changing Consumer Behavior:** Modern consumers prioritize convenience, price comparisons, and personalized recommendations when shopping.
- **Limited Access to Data Analytics:** Retailers lack insights into pricing trends, demand forecasting, and customer preferences, leading to suboptimal decision-making.

2.2 Market Opportunities

- **Growth of Digital Commerce:** The increasing use of mobile apps and online platforms presents an opportunity for local retailers to digitize their operations.
- **Hybrid Shopping Trends:** Consumers prefer a mix of online browsing and offline purchases, allowing local retailers to attract in-store visits through online engagement.
- **Personalized Shopping Experience:** AI-driven insights can help retailers offer targeted promotions and enhance customer engagement.

3. Product Introduction

TechNest is an AI-powered platform designed to bridge the gap between local retailers and online consumers by providing real-time price comparisons, personalized product recommendations, and seamless demo booking services. The platform enhances price transparency, boosts local retailer sales, and improves the shopping experience for customers.

3.1 Key Features

- **Real-time Price Comparison:** Consumers can compare local store prices with online platforms to make informed purchase decisions.
- **Product Discovery & Recommendations:** AI-driven insights help consumers find the best deals and relevant products based on their preferences.
- **Retailer Dashboard:** Provides retailers with data analytics, inventory management, and customer insights.
- **Seamless Demo Booking:** Allows customers to book in-store product demos directly from the platform.
- **Secure Transactions:** Integrated payment options ensure smooth and secure purchasing.

3.2 Competitive Advantages

- **Localized Pricing Insights:** Unlike generic e-commerce platforms, TechNest focuses on local store price comparisons.
- **Retailer-Focused Solutions:** Helps small retailers improve their online visibility and increase sales.
- **AI-Driven Personalization:** Enhances the shopping experience by offering tailored recommendations and insights.

- **Hybrid Shopping Support:** Enables consumers to browse online and purchase offline, blending digital convenience with in-store engagement.

4. Business Need Assessment

The retail industry, particularly in the electronics sector, faces multiple challenges due to increasing competition from e-commerce platforms and shifting consumer behaviors. Small and mid-sized retailers struggle with digital transformation, leading to lower sales and reduced visibility. On the consumer side, shoppers seek better deals, price transparency, and a seamless online-to-offline experience.

TechNest aims to bridge this gap by providing a platform that benefits both consumers and retailers through price comparison, product discovery, and AI-driven recommendations.

4.1 Market Challenges

1. Lack of Digital Presence for Retailers

- Many local retailers do not have an online presence, making it difficult for them to compete with e-commerce platforms.
- Limited digital marketing capabilities reduce their ability to attract customers.

2. Price Transparency Issues

- Consumers often find it difficult to compare local store prices with online marketplaces.
- Retailers lack real-time insights into competitive pricing strategies, making it harder to adjust their pricing dynamically.

3. Changing Consumer Shopping Behavior

- Modern shoppers prefer researching products online before making a purchase.
- The growing trend of “webrooming” (checking online before buying offline) requires retailers to be present digitally.

4. Limited Customer Engagement & Retention

- Without data-driven insights, retailers struggle to engage customers effectively.
- Lack of personalized promotions and recommendations leads to lower conversion rates.

5. Operational Inefficiencies

- Retailers often lack access to inventory management, sales forecasting, and analytics tools.
- Manual processes result in poor demand forecasting, stock mismanagement, and increased costs.

4.2 TechNest's Solutions

1. Retailer Digital Enablement

- Provides a centralized dashboard for retailers to list their products, manage inventory, and gain customer insights.
- Subscription-based model ensures affordability for small businesses.

2. AI-Driven Price Comparison & Recommendations

- Helps consumers compare local prices with online platforms in real time.
- Enables retailers to adjust their pricing strategy based on competitor insights.

3. Seamless Online-to-Offline Shopping Experience

- Consumers can browse and book in-store product demos.
- Encourages foot traffic to physical retail stores.

4. Customer Engagement & Loyalty Building

- Personalized recommendations improve conversion rates.
- Loyalty programs and targeted promotions encourage repeat purchases.

5. Analytics & Business Intelligence

- Retailers gain insights into consumer preferences, best-selling products, and sales trends.
- AI-driven analytics help in optimizing stock levels and maximizing revenue.

5. Target Audience

TechNest is designed to cater to both **consumers** and **retailers**, ensuring a seamless shopping experience while helping businesses thrive in the digital space. Our primary target audience includes:

5.1 Consumers

1. Tech-Savvy Shoppers

- Consumers who extensively research products online before purchasing.
- Prefer price comparisons and AI-driven recommendations to find the best deals.
- Actively engage in both online and offline shopping.

2. Working Professionals & Students

- Individuals who seek convenience and quick access to the best available deals.
- Prefer pre-booking in-store demos before making high-value purchases.

3. Bargain Hunters & Deal Seekers

- Users looking for competitive prices and discounts from both online and local retailers.
- Regularly check for price variations and offers before making a purchase.

4. **Brand-Conscious Buyers**

- Consumers who prioritize brand reputation, quality, and product authenticity.
- Willing to pay a premium for verified products but still seek competitive pricing.

5.2 Retailers

1. **Small & Mid-Sized Retailers**

- Local electronic store owners looking to expand their digital presence.
- Need an affordable platform to attract more customers and increase foot traffic.

2. **Enterprise & Large-Scale Retailers**

- Established retailers who want to enhance price competitiveness and gain deeper market insights.
- Require AI-driven analytics to optimize pricing and customer engagement strategies.

3. **E-commerce & Hybrid Retailers**

- Businesses that operate both online and offline, looking to bridge the gap between digital and in-store shopping.
- Seek data-driven insights to maximize conversions across multiple sales channels.

4. **Retailers Focused on Customer Experience**

- Businesses that emphasize personalized shopping experiences, demo bookings, and high customer engagement.
- Want to integrate smart recommendations and loyalty programs into their sales strategy.

6. ML Model Development

The TechNest platform integrates **AI-driven product recommendations** to enhance the shopping experience for consumers while providing valuable insights for retailers. The recommendation system is based on **K-Nearest Neighbors (KNN)** and utilizes product price and store location to find the most relevant products for users.

The architecture balances efficiency and computational performance by leveraging a **feature-based similarity model**. Below is a detailed explanation of each component:

6.1 Input Features

- **Product Price:** Numeric value representing the cost of a product.
- **Store Location:** Categorical data converted into a one-hot encoded feature.
- **User Preferences:** Input variables like price range and preferred store locations.

6.2 Feature Processing

Before training the model, the following preprocessing steps are applied:

- **Normalization:** Price values are normalized for uniform scaling.
- **One-Hot Encoding:** Categorical variables (store location) are converted into binary indicators.
- **Feature Vector Formation:** Each product is represented as a vector combining price and location features.

6.3 K-Nearest Neighbors (KNN) Algorithm

The recommendation system utilizes **KNN**, which is a simple yet effective technique for similarity-based retrieval.

KNN-Based Product Recommendation Algorithm

Step 1: Feature Extraction

- Extracts product price and store location as key recommendation features.
- One-hot encoding is applied to location data to ensure compatibility with distance calculations.

Step 2: Model Training

- The dataset is stored in a structured form with feature vectors.
- The **KNN model** computes Euclidean distances between the user's preferred price/location and available products.

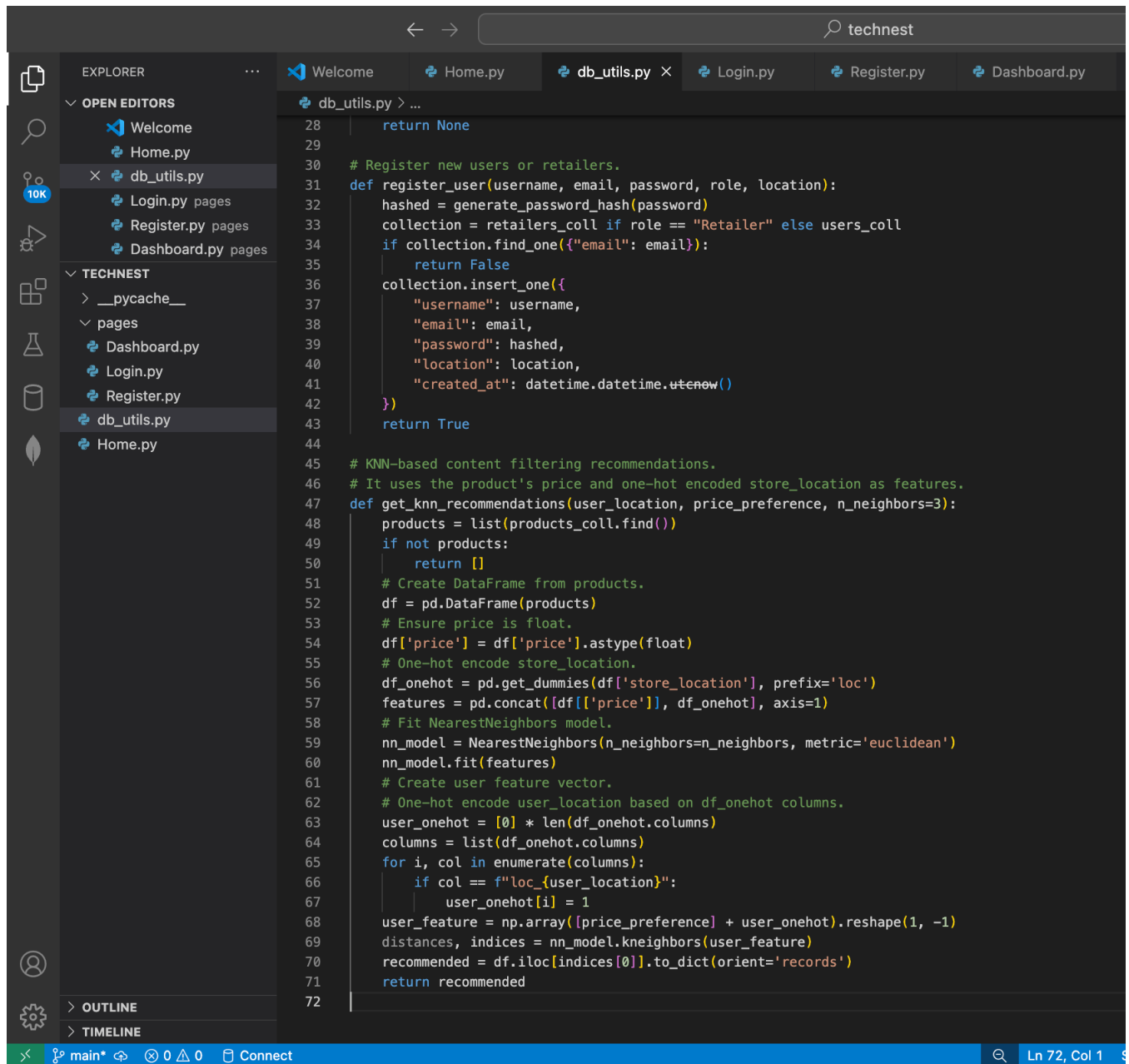
$$d(U, P) = \sqrt{\sum_{i=1}^n (U_i - P_i)^2}$$

where **n** represents the total number of features (price, location encodings).

Step 3: Recommendation Retrieval

- For a given user input (price range & location), the algorithm retrieves the **K** nearest products based on feature similarity.
- The **top 3 most relevant products** are displayed.

6.4 Model Implementation



The screenshot shows a code editor with a dark theme. The Explorer panel on the left shows a project named 'technest' with a file structure including 'db_utils.py'. The main editor window displays the code for 'db_utils.py'. The code includes a function 'register_user' and a function 'get_knn_recommendations'. The 'register_user' function takes 'username', 'email', 'password', 'role', and 'location' as arguments and returns 'None' or 'True'. The 'get_knn_recommendations' function takes 'user_location', 'price_preference', and 'n_neighbors' as arguments and returns a 'recommended' product. The code uses MongoDB for database operations and scikit-learn for KNN-based content filtering.

```
28     return None
29
30 # Register new users or retailers.
31 def register_user(username, email, password, role, location):
32     hashed = generate_password_hash(password)
33     collection = retailers_coll if role == "Retailer" else users_coll
34     if collection.find_one({"email": email}):
35         return False
36     collection.insert_one({
37         "username": username,
38         "email": email,
39         "password": hashed,
40         "location": location,
41         "created_at": datetime.datetime.utcnow()
42     })
43     return True
44
45 # KNN-based content filtering recommendations.
46 # It uses the product's price and one-hot encoded store_location as features.
47 def get_knn_recommendations(user_location, price_preference, n_neighbors=3):
48     products = list(products_coll.find())
49     if not products:
50         return []
51     # Create DataFrame from products.
52     df = pd.DataFrame(products)
53     # Ensure price is float.
54     df['price'] = df['price'].astype(float)
55     # One-hot encode store_location.
56     df_onehot = pd.get_dummies(df['store_location'], prefix='loc')
57     features = pd.concat([df[['price']], df_onehot], axis=1)
58     # Fit NearestNeighbors model.
59     nn_model = NearestNeighbors(n_neighbors=n_neighbors, metric='euclidean')
60     nn_model.fit(features)
61     # Create user feature vector.
62     # One-hot encode user_location based on df_onehot columns.
63     user_onehot = [0] * len(df_onehot.columns)
64     columns = list(df_onehot.columns)
65     for i, col in enumerate(columns):
66         if col == f"loc_{user_location}":
67             user_onehot[i] = 1
68     user_feature = np.array([price_preference] + user_onehot).reshape(1, -1)
69     distances, indices = nn_model.kneighbors(user_feature)
70     recommended = df.iloc[indices[0]].to_dict(orient='records')
71     return recommended
72
```


Customer Dashboard:

```
# -----
# Customer Dashboard
# -----
elif role == "Customer":
    st.subheader("Search for Products")
    query = st.text_input("Enter product name")

    if st.button("Search"):
        results = list(products_coll.find({"name": {"$regex": query, "$options": "i"}}))
        if results:
            for prod in results:
                st.markdown("----")
                st.write(f"**{prod['name']}**")
                st.write(prod["description"])
                st.write(f"Your Price: ${prod['price']}")
                amazon_price = get_amazon_price(prod["name"])
                st.write(f"Amazon Price: ${amazon_price}")
                if prod.get("image"):
                    st.image(prod["image"], width=150)
                # Book Demo button
                if st.button("Book Demo", key=str(prod["_id"])):
                    # In real implementation, send a message to the retailer.
                    st.success("Demo request sent to retailer!")
                col1, col2 = st.columns(2)
                with col1:
                    if st.button("Buy", key="buy"+str(prod["_id"])):
                        st.info("Buy functionality not implemented.")
                with col2:
                    if st.button("Add to Cart", key="cart"+str(prod["_id"])):
                        st.info("Add to Cart functionality not implemented.")
            else:
                st.info("No products found.")

    st.subheader("Recommended Products")
    # Dummy recommendation: randomly sample up to 3 products.
    total_count = products_coll.count_documents({})
    if total_count:
        recommendations = random.sample(list(products_coll.find()), min(3, total_count))
        for rec in recommendations:
            st.markdown("----")
            st.write(f"**{rec['name']}**")
            st.write(f"Price: ${rec['price']} | Location: {rec['store_location']}")
            if rec.get("image"):
                st.image(rec["image"], width=150)

    st.subheader("Nearby Retailers")
    # Dummy implementation: Show retailers matching the customer's location.
    user = users_coll.find_one({"_id": ObjectId(user_id)})
    user_location = user.get("location", "")
    nearby = list(retailers_coll.find({"store_location": user_location}))
    if nearby:
        for ret in nearby:
            st.write(f"{ret['username']} - {ret['store_location']}")
    else:
        st.info("No nearby retailers found.")
```

Retailers dashboard

```
# -----
# Retailer Dashboard
# -----
if role == "Retailer":
    st.subheader("Retailer Dashboard: Add Product")

    name = st.text_input("Product Name")
    description = st.text_area("Description")
    price = st.number_input("Price", min_value=0.0, format="%.2f")
    store_location = st.text_input("Store Location")
    uploaded_file = st.file_uploader("Upload Product Image", type=["png", "jpg", "jpeg"])

    if st.button("Add Product"):
        image_data = None
        if uploaded_file is not None:
            image_data = uploaded_file.read() # read image bytes
        products_coll.insert_one({
            "retailer_id": user_id,
            "name": name,
            "description": description,
            "price": price,
            "store_location": store_location,
            "image": image_data, # storing binary image data; in production, consider saving fil
            "created_at": datetime.datetime.utcnow()
        })
        st.success("Product added successfully!")

    st.subheader("Your Products")
    my_products = list(products_coll.find({"retailer_id": user_id}))
    for prod in my_products:
        st.write(f"*{prod['name']}* - ${prod['price']} | {prod['store_location']}")
        st.write(prod["description"])
        if prod.get("image"):
            st.image(prod["image"], use_column_width=False, width=150)
```

6.5 Model Components

1. Input Layer

- **Feature Vector Input:** Contains price and location features.
- **Shape:** $[N, F][N, F][N, F]$, where **N** is the number of products and **F** is the number of features (price, location encodings).

2. Feature Extraction & Similarity Calculation

- **Distance Metric:** Euclidean Distance.
- **Computes nearest neighbors** based on feature similarity.

3. Recommendation Engine

- Identifies **top K similar products** based on pricing and location.
- Outputs relevant product details to the user.

6.6 Future Enhancements

1. Deep Learning for Personalization

- Implementing **Neural Networks** to analyze user behavior and generate dynamic recommendations.

2. Sentiment Analysis for Product Insights

- Extracting insights from customer reviews to refine recommendations.

3. Hybrid Recommendation System

- Combining **content-based filtering (KNN)** with **collaborative filtering** to improve accuracy.

7. Product Prototype Report

TechNest follows a structured **operational workflow** that ensures seamless interaction between users, retailers, and the recommendation engine. The platform integrates **web and mobile interfaces**, a backend API server, a robust database system, and an AI-powered recommendation engine to enhance the shopping experience.

1. User Device (Web / Mobile App)

- The primary interface where consumers interact with the platform.
- Users can search for products, compare prices, receive AI-powered recommendations, and book in-store demos.
- Available on both web browsers and mobile applications.

2. Frontend (React.js / React Native)

- Built using React.js (for web) and React Native (for mobile apps).
- Provides a smooth and responsive user experience, ensuring easy navigation and intuitive design.

- Handles user authentication, product search, and displaying recommendations.

3. API Server (Node.js with Express.js)

- Acts as a middleware between the frontend and backend components.
- Manages:
 - User authentication (login, registration).
 - Retailer data (product listings, store details).
 - Price comparisons by retrieving and processing product price details.
- Ensures secure data transactions between the frontend, database, and AI model.

4. User Database (MongoDB)

- Stores user-related information, including:
 - Profile details (name, email, preferences).
 - Search history and past purchases.
 - Reviews and product ratings.
 - Enables personalized recommendations based on previous interactions.

5. Store Database (MongoDB)

- Maintains data for local retailers and their products.
- Stores:
 - Retailer business details (name, location, contact info).
 - Product inventory (product names, prices, availability).
 - Sales records and order tracking.
- Provides real-time updates on product availability.

6. Product Data

- Centralized storage for all product-related details, including:
 - Product specifications (brand, category, features).
 - Retail and online marketplace prices (local store price vs. Amazon price).
 - Stock availability across different retailers.
- Helps in generating accurate price comparisons.

7. Machine Learning Service

- Implements AI-driven product recommendations using:
 - K-Means Clustering: Groups similar products based on price range and features.
 - K-Nearest Neighbors (KNN): Finds the best-matching products based on user preferences.
- Ensures a personalized shopping experience by suggesting relevant products.

8. Recommendation Engine & Price Comparison

- Core AI system that enhances decision-making for users.
- Functions:
 - Analyzes user behavior to recommend the most suitable products.
 - Compares prices across multiple retailers to suggest the best deals.
 - Provides alternative product suggestions if a preferred product is unavailable.

9. Local Stores Portal

- Retailer-facing dashboard where store owners can:
 - List new products and update pricing.
 - Monitor sales and customer interactions.
 - Respond to customer queries and offer promotions.
- Ensures easy digital transformation for local retailers, improving their online visibility.

7.1 Advantages of the Prototype

Feature	Benefit
AI-Driven Product Discovery	Users find the best-matching products with minimal effort.
Real-Time Price Comparison	Ensures customers get the best deals across retailers.
Retailer Digital Enablement	Helps local stores establish an online presence.
Seamless Integration	Works across web and mobile platforms.
User Behavior Learning	System improves recommendations over time.

7.2 Future Enhancements

- **Integration with Augmented Reality (AR)** for virtual product testing.
- **Voice-Assisted Shopping** using AI-driven chatbots.
- **Dynamic Pricing Adjustments** for retailers based on demand analysis.

8. TechNest Business Model Report

TechNest employs a diversified revenue approach to ensure **scalability and long-term profitability** while offering value to both **customers and retailers**.

8.1 Revenues

1. Freemium Model

- Free access to basic product search and price comparison features.
- Encourages adoption and builds an initial user base.

2. Premium Subscription Plans

- Retailers pay a monthly/yearly fee to list their products and access advanced analytics.

- **Subscription Tiers:**
 - **Basic Plan:** Allows retailers to list a limited number of products.
 - **Pro Plan:** Provides AI-driven pricing insights, enhanced product visibility, and customer analytics.
 - **Enterprise Plan:** Offers customized store branding, priority listing, and promotional tools.
- 3. **Commission-Based Sales**
 - A small percentage commission is charged on each successful transaction completed through the platform.
- 4. **Advertisement & Sponsored Listings**
 - Retailers can pay for featured product placements to boost their store visibility.
 - Targeted Ads: Personalized ads displayed based on customer behavior.
- 5. **Affiliate Marketing**
 - Comparison with Amazon & other marketplaces generates revenue from referral sales.

8.2 Subscription Plans for Retailers

Plan	Features	Pricing
Basic	List up to 10 products, Basic analytics	₹500/month
Pro	AI-based price insights, Priority listing, 50+ products	₹1500/month
Enterprise	Unlimited products, Advanced analytics, Store branding	₹5000/month

8.3 Financial Assumptions

1. **Product Pricing and Cost Assumptions:**
 - Product listing fee: ₹500 per retailer per month.
 - Cost of operation (server, maintenance, marketing): ₹30,000/month.
2. **Market Penetration Strategy:**
 - Initial penetration: 0.005% of retailers.
 - Scaling up to 0.05% penetration by 2029.

8.4 Revenue Projection & Financial Equation

Using the revenue equation:

$$\text{Revenue} = (\text{Subscription Fee} \times x) - \text{Operational Cost}$$

where x = Number of subscribing retailers.

<i>Year</i>	<i>Market Size (Billion USD)</i>	<i>Penetration (%)</i>	<i>Retailers Subscribed (Million)</i>	<i>Revenue (Million ₹)</i>
2025	320.50	0.005%	0.01603	8.01
2026	348.80	0.01%	0.03488	17.44
2027	375.20	0.02%	0.07504	37.52
2028	398.90	0.03%	0.11967	59.83
2029	420.30	0.05%	0.21015	105.08

Example Calculations

For **2025**:

Retailers Subscribed= $320.50B \times 0.005\% = 0.01603M$ (16,030 retailers)

Revenue= $(500 \times 16030) - 30000 = ₹8,01,50,000$

For **2029**:

Retailers Subscribed= $420.30B \times 0.05\% = 0.21015M$ (210,150 retailers)

Revenue= $(500 \times 210150) - 30000 = ₹1,05,08,50,000$

8.5 Insights and Recommendations

1. Growth Trends

- The market demand for AI-driven retail platforms is growing steadily.
- Need to scale early to capture market share before saturation occurs.

2. Break-Even Analysis

- Break-even occurs when revenue covers operational costs:

$$500x - 30000 = 0$$

Solving for x :

$$x = 60 \text{ (minimum subscribers per month to break even)}$$

3. Opportunities for Expansion

- Integration with major e-commerce platforms for cross-selling.
- Partnering with retail businesses to increase market reach.
- Leveraging AI & predictive analytics to enhance user engagement.

9. Conclusion

This report provides a structured approach to our business model, financial sustainability, and targeted market segmentation. By leveraging data-driven strategies, we aim to establish TechNest as a competitive player in the market. Further iterations and refinements will be based on market feedback and financial performance.