Perfect timing 🔥 — since your **research paper viva** is **today**, and your project is the **E-commerce Product Return Prediction Dashboard** (Flask + ML + XGBoost),
here's a complete **Viva Q&A Sheet (customized exactly for your interface, dataset, and models)**.

---

🎓 **Research Paper Viva – Quick Revision Sheet**

*(Project: Product Return Prediction Dashboard using Machine Learning)*

---

🔹 **1. What is your project about?**

Our project focuses on predicting whether a product will be returned or not based on customer and transaction data.
We built a web-based **Return Prediction Dashboard** using **Flask**, which visualizes sales, returns, and model performance.

---

🔹 **2. Which dataset did you use?**

We used the **ecommerce_sales_34500.csv** dataset.
It includes features such as:

- region (customer location)
- category (product type)
- payment_method (online or COD)
- customer_gender
- order_date, quantity, price, etc.
- Target variable: returned (Yes/No)

---

🔹 **3. What preprocessing steps did you perform?**

- Converted order_date → extracted **Year, Month, Day, Weekday**
- Converted returned → binary (1 for Yes, 0 for No)
- Applied **Label Encoding** for categorical columns (region, category, payment_method, gender)
- Used **SMOTE oversampling** to handle class imbalance (many "No Return" vs fewer "Return")

---

🔹 **4. What machine learning models did you use?**

We implemented three models:

1. **Logistic Regression** → baseline linear model
2. **Random Forest** → ensemble model for nonlinear data

3. **XGBoost** → advanced gradient boosting model (performed best)

---

🔹 **5. Why did you choose XGBoost?**

- XGBoost handles **imbalanced datasets** better than other models.
- It combines multiple weak learners to improve recall and accuracy.
- It gave the **best recall (~31%)** in our results, detecting more true returns.

---

🔹 **6. What problem did you face with Random Forest?**

Random Forest achieved **high accuracy (~0.94)** but **0 recall**,
meaning it predicted almost all cases as "No Return".
This happened because of **class imbalance** — the model favored the majority class.

---

🔹 **7. What is SMOTE, and why did you use it?**

SMOTE stands for **Synthetic Minority Oversampling Technique**.
It artificially creates new samples for the minority class ("Returns") to balance the dataset.
This helps models learn better and improves recall and F1-score.

---

🔹 **8. What evaluation metrics did you use and why?**

We used:

- **Accuracy** → Overall correctness
- **Precision** → How many predicted returns were actually returns
- **Recall** → How many actual returns were correctly identified
- **F1-Score** → Balance between precision and recall
  Because our dataset was imbalanced, **Recall and F1-Score** were more important than Accuracy.

---

🔹 **9. What were your model results?**

| Model | Accuracy | Precision | Recall | F1-Score |
|---|---|---|---|---|
| Logistic Regression | ~0.54 | 0.06 | 0.52 | 0.11 |
| Random Forest | ~0.94 | 0.00 | 0.00 | 0.00 |
| XGBoost | ~0.73 | 0.07 | 0.31 | 0.11 |

✅ **XGBoost performed best overall** because it captured more true returns.

---

🔹 **10. What graphs are shown in your dashboard?**

Our Flask dashboard displays:

- 📊 Return Rate by Year
- 📦 Orders by Category
- 💳 Orders by Payment Method
- 🌍 Returns by Region
- 🤖 Model Performance Comparison (Accuracy, Precision, Recall, F1)

---

🔹 **11. What technologies did you use?**

- **Frontend:** HTML, CSS
- **Backend:** Flask (Python)
- **Libraries:** Pandas, Matplotlib, Scikit-learn, XGBoost, Imblearn (SMOTE)
- **IDE:** VS Code
- **Visualization:** Rendered charts via base64 images in Flask template

---

🔹 **12. What were your key findings?**

- Return probability depends on **Category, Payment Method, and Region**.
- **COD (Cash on Delivery)** orders had a higher return rate.
- **Fashion & Electronics** categories were returned more often.
- **XGBoost** gave the best predictive balance.

---

🔹 **13. What are the limitations of your project?**

- Dataset is **imbalanced** (few returns).
- Limited customer behavior data (no reasons for returns).
- Precision remains low — model predicts some false positives.

---

🔹 **14. What are possible future improvements?**

- Use **cost-sensitive learning** or **threshold tuning**.
- Collect **more return samples** or add **customer review sentiment**.

- Add **real-time prediction API** for live e-commerce integration.
- Enhance dashboard interactivity (filters by region/category).

---

🔹 **15. What are your final conclusions?**

- **XGBoost** was the most effective model for detecting returns (best recall).
- **Random Forest** had highest accuracy but failed in recall.
- **Logistic Regression** served as baseline.
- **SMOTE** improved minority detection.
- Dashboard provides clear insights for business decision-making.

---

🔹 **16. Can you explain your dashboard interface?**

- **Top Navbar:** Title and filters (Year, Region, Category).
- **Left Section:** Filter options.
- **Right Section:** Dynamic graphs + Model comparison table.
- **Output:** Updates based on user input (Flask form submission).

---

🔹 **17. What did you learn from this project?**

- Handling imbalanced datasets with SMOTE.
- Comparing ML algorithms effectively.
- Integrating ML with Flask for real-time visualization.
- Importance of precision vs recall in real business applications.

---

✅ **Pro Tip for Viva:**
If they ask “Which model will you deploy?”, answer:

“We would deploy **XGBoost** because it balances recall and overall performance better, making it more reliable for identifying potential returns in real-world applications.”

---

Would you like me to make this into a **1-page printable PDF Viva Sheet** (so you can quickly glance at it before your viva)?