🎓 VIVA QUESTIONS & ANSWERS — WEEK 1 (Exploration)

1️⃣ What is the main objective of your project?

Answer:
The objective is to predict whether a patient has cardiovascular disease using clinical features such as age, blood pressure, cholesterol, glucose levels, and lifestyle habits.

2️⃣ What type of machine learning problem is this?

Answer:
It is a **binary classification** problem because the target variable `cardio` has two values: 0 or 1.

3️⃣ What dataset are you using?

Answer:
I am using the **Cardiovascular Disease Dataset** from Kaggle, which contains 70,000 patient records with 12 features.

4️⃣ What is the meaning of the target variable?

Answer:
`cardio = 0` → No cardiovascular disease
`cardio = 1` → Cardiovascular disease present

5️⃣ Why is the age column in days?

Answer:

The dataset stores age in days for more accuracy.

We convert it to **years** for meaningful interpretation.

6️⃣ Which columns are numerical and categorical?

Answer:

Numerical: age, height, weight, ap_hi, ap_lo, bmi

Categorical: gender, cholesterol, gluc, smoke, alco, active

7️⃣ Did your dataset contain missing values?

Answer:

No, the dataset did not contain missing values.

8️⃣ What initial patterns did you observe?

Answer:

- Blood pressure and BMI have outliers
- Age is in days
- Target variable is moderately balanced
- Cholesterol and glucose levels influence cardio risk

9️⃣ Why did you draw histograms in Week-1?

Answer:

To observe data distribution and detect outliers or skewness in numerical features.

🔟 Why did you plot the correlation heatmap?

Answer:

To see relationships between features and identify which features strongly relate to the target.

🎓 VIVA QUESTIONS & ANSWERS — WEEK 2 (Cleaning & Preprocessing)

1️⃣ Why is data cleaning important?

Answer:
Cleaning removes incorrect, inconsistent, or unrealistic values, improving model accuracy and reducing noise.

2️⃣ Why did you convert age from days to years?

Answer:
Years are more interpretable for humans and more meaningful for medical analysis.

3️⃣ What kind of outliers did you remove?

Answer:

- Wrong BP values: `ap_hi < ap_lo`
- Extremely high or low height
- Very unrealistic weights (e.g., <30 or >200 kg)

4️⃣ What is BMI and why did you create it?

Answer:
BMI = weight / height^2
It is an important indicator for heart disease risk, so we added it as a new feature.

5️⃣ Why did you scale the numerical features?

Answer:
Scaling brings features to the same range, improving gradient descent performance and model stability.

6️⃣ Which scaler did you use? Why?

Answer:
I used **StandardScaler**, which transforms data to mean = 0 and standard deviation = 1.
It works well for logistic regression.

7️⃣ What is EDA? Why is it done?

Answer:
Exploratory Data Analysis helps understand patterns, detect outliers, identify correlations, and guide modeling decisions.

8️⃣ Why did you draw boxplots?

Answer:
To compare distributions of features like age or BMI across the target classes (`cardio` 0 and 1).

9️⃣ Why did you create countplots for categorical features?

Answer:
To observe how cholesterol, glucose, smoking, alcohol, and activity levels differ between disease and non-disease groups.

🔟 Why did you save the cleaned dataset?

Answer:
To use it directly in Week-3 for training the ML model without repeating cleaning steps.

🎓 BONUS — HIGH-SCORING ANSWERS

⭐ Why did you choose this dataset?

Answer:
It has both clinical and lifestyle features, making it suitable for building an interpretable medical prediction model.

⭐ What is the data size?

Answer:
About 70,000 rows and 12 features.

⭐ Why is preprocessing important before ML?

Answer:
Because raw data contains noise, invalid values, and different scales. ML models perform poorly without proper preprocessing.

🎤 VIVA ANSWERS (Short & Clear)

✅ 1. What are weights in Logistic Regression?

Weights are the importance values learned by the model for each feature.
They tell how much each feature contributes to the prediction.

👉 If a weight is **positive**, it increases disease probability.
👉 If a weight is **negative**, it decreases disease probability.
👉 Larger weight = more influence.

Example viva answer:

> "Weights determine how strongly each feature affects the final prediction.
> For example, if the weight for cholesterol is high, it means cholesterol is an important factor for predicting cardiovascular disease."

✅ 2. What is bias?

Bias is a constant value added to shift the decision boundary.

Even if all inputs are zero, the model can still make a prediction using bias.

Example viva answer:

> "Bias helps the model adjust the output even when all features are zero.
> It shifts the prediction curve and improves accuracy."

✅ 3. Why do we save weights and bias?

Because **weights + bias = trained model**.
Without them, the model cannot make predictions.

Example viva answer:

> "After training, the weights and bias represent the learned knowledge of the model.
> We save them so we can use the trained model later in the Flask web app without retraining."

✅ 4. Why do we save them in `.npy` files?

- `.npy` is the best format for saving **NumPy arrays**
- Very fast to load
- Stores exact values without losing precision

Example viva answer:

> "Weights and bias are NumPy arrays, and `.npy` is the most efficient format to store them.
> It loads quickly and preserves all values accurately, which is important for deployment."

✅ 5. Why two separate files? (`weights.npy` and `bias.npy`)

Because weights and bias have different shapes.

- weights = array of many values
- bias = a single number

Example viva answer:

> "Weights and bias have different shapes, so saving them separately avoids confusion and makes it easy to load and use them in Flask."

✅ 6. Why Logistic Regression for this dataset?

- Binary classification problem
- Medical dataset → interpretability needed
- Works well with numerical & categorical features
- Lightweight and fast

Example viva answer:

> "Since our dataset has a binary target (0 or 1), Logistic Regression is the most suitable.
> It is simple, interpretable, and effective for medical predictions."

✅ 7. Why implement Logistic Regression from scratch?

Because the assignment requires:

> "Implement selected algorithm without library."

Example viva answer:

> "We implemented Logistic Regression using NumPy to understand how the algorithm works internally — gradient descent, loss calculation, and weight updates — instead of depending on scikit-learn."

✅ 8. Can we use scikit-learn?

Yes, only for comparison.
Main model must be from scratch.

Example viva answer:

> "I used scikit-learn only to compare performance with my scratch model.
> The main model was implemented manually using NumPy as required."

✅ 9. What does the Sigmoid function do?

It converts numbers into probabilities between 0 and 1.

Example viva answer:

> "Sigmoid maps the linear model output into a probability value, which helps classify whether a patient has cardiovascular disease or not."

🎯 10. What does the loss function represent?

It measures how wrong the model's predictions are.

Example viva answer:

> "I used Binary Cross Entropy as the loss.
> Lower loss means better predictions."

🎉 SUMMARY — Viva Notes (Super Short)

You can memorize this:

- **Weights** → importance of features
- **Bias** → adjusts output
- **.npy** → saves NumPy arrays safely
- **Scratch model** → uses gradient descent
- **Sigmoid** → converts to probability
- **Logistic Regression** → best for binary classification
- **Sklearn** → only for comparison
- **Two files** → shapes are different (matrix & scalar)

🎓 VIVA QUESTIONS & ANSWERS — WEEK 3 (Model Creation & Evaluation)

1️⃣ What algorithm did you choose and why?

Answer:
I chose **Logistic Regression** because the dataset has a **binary target** (0 or 1), and logistic regression is simple, interpretable, and works very well for medical predictions.

2️⃣ Why did you implement the algorithm from scratch?

Answer:
The project requirement was to implement an algorithm **without using any ML library**, so I used NumPy to create logistic regression manually.
This helped me understand how gradient descent, sigmoid, and loss functions work internally.

3️⃣ What is the sigmoid function? Why do we use it?

Answer:
Sigmoid converts any value into a probability between 0 and 1.
It's required for binary classification.

4️⃣ What is the loss function used?

Answer:
I used **Binary Cross Entropy Loss**, which measures how well the predicted probabilities match the actual labels.

5️⃣ What is gradient descent?

Answer:
Gradient Descent is an optimization algorithm that updates the weights and bias step-by-step to reduce the loss.

6️⃣ What are weights and bias?

Answer:
Weights represent importance of each feature.
Bias allows shifting the decision boundary.
Together they determine the prediction.

7️⃣ Why do we save weights and bias?

Answer:
Weights and bias are the **trained model parameters**.
Once saved, we can use them in Week-4 Flask app without retraining the model.

8️⃣ Why save them as `.npy` files?

Answer:
`.npy` is the most efficient format to store NumPy arrays.
It loads fast, preserves precision, and is easy to use in Flask.

9️⃣ Did you compare your scratch model with scikit-learn?

Answer:
Yes. I trained a `LogisticRegression` model using scikit-learn to compare accuracy and confirm that my scratch implementation works correctly.

🔟 What metrics did you use to evaluate the model?

Answer:

- Accuracy
- Precision
- Recall
- F1-Score
- Confusion Matrix
- ROC Curve and AUC

⭐ 11. How did you check for overfitting or underfitting?

Answer:
I compared training and testing accuracy.
If training accuracy is much higher than test accuracy, it indicates overfitting.
In my case, both were similar, meaning the model generalized well.

⭐ 12. What hyperparameters did you tune?

Answer:

- Learning rate
- Number of iterations
  Both affect the speed and convergence of gradient descent.

🎓 VIVA QUESTIONS & ANSWERS — WEEK 4 (Flask App + Model Deployment)

1️⃣ What did you do in Week-4?

Answer:
I created a Flask web application that loads the saved model weights and bias, takes user input, applies preprocessing, and predicts the heart disease risk.

2️⃣ Why did you choose Flask?

Answer:
Flask is lightweight, easy to use, and perfect for small machine-learning deployment projects.
It allows fast creation of web forms and API routes.

3️⃣ How does your Flask app make predictions?

Answer:

1. Takes input from HTML form
2. Converts it into a feature vector
3. Loads saved weights and bias
4. Applies sigmoid function
5. Returns disease probability

4️⃣ What files does your Flask app use?

Answer:

- `app.py` → main server file
- `model_utils.py` → loads weights & performs prediction
- `index.html` → input form
- `result.html` → displays prediction
- `.npy` files → saved model parameters

5️⃣ How do you load the saved model in Flask?

Answer:

Using NumPy:

```
weights = np.load("logistic_weights.npy")
bias = np.load("logistic_bias.npy")
```

6️⃣ What is the role of `model_utils.py`?

Answer:

It contains the prediction logic:

- Load weights
- Scale features
- Compute `z = wx + b`
- Apply sigmoid
- Return probability & label

7️⃣ What preprocessing do you apply inside Flask?

Answer:

Scaling of numerical features using the saved scaler (`StandardScaler`) from Week-2.

Then combine numeric + categorical features into a final input vector.

8️⃣ How does your HTML form pass data to Flask?

Answer:

Using a POST request:

```
<form action="/predict" method="POST">
```

9️⃣ What is the `/predict` route?

Answer:

It collects form data, calls the prediction function, and returns the result to the user.

🔟 Why is deployment important?

Answer:
Deployment allows others to use the ML model through a simple web interface without running Jupyter Notebook.

🎉 BONUS – Extra Viva Answers (High Scoring)

⭐ What are the challenges in deploying a machine learning model?

Answer:

- Preprocessing consistency
- Loading trained parameters
- Handling user inputs
- Ensuring fast inference
- Environment dependencies

⭐ Why is model interpretability important in medical predictions?

Answer:
Doctors need to understand which features influence the prediction.
Logistic regression provides weights that reflect feature importance.