Team Roles (Suggestion - Adapt as needed):

- **Speaker 1 (Intro & Overview):** Covers Slides 1-5.
- **Speaker 2 (Live Demo):** Covers the application demonstration (triggered after Slide 10).
- **Speaker 3 (Technical Details):** Covers Slides 12-30 (Tech, SWE, Design, SOLID).
- **Speaker 4 (Traceability & Conclusion):** Covers Slides 31-39.

Presentation Script & Flow (20 Minutes)

(0:00 - 2:00) Intro & Overview - Speaker 1

- **(Slide 1: Title Slide)**
    - "Good morning/afternoon everyone. We are SCMA Group 2: [Briefly mention names - Athithiya, Benjamin, Xuehao, Javier, Tristan, Yash]. Today we're presenting FetchMeHome."

- **(Slide 2: Problem)**
    - "Our project addresses two key challenges in pet welfare: firstly, pets missing out on homes because owners struggle to meet their needs, leading to potential abandonment or euthanasia, and secondly, the lack of an effective digital platform for owners to reconnect with the community when a pet goes missing."

- **(Slide 3: Reconnect Question)**
    - "This led us to ask: What if we could leverage technology to efficiently reconnect lost pets with owners and find loving homes for adoptable pets?"
- **(Slide 4: Solution - FetchMeHome)**
    - "Our solution is FetchMeHome, a MERN stack web application. It's a community-driven platform designed to connect adopters with pets needing homes and to help safely retrieve lost pets through public assistance."

- **(Slide 5: Key Features)**
    - "FetchMeHome integrates several key features, including secure Authentication, managing Adoption and Lost Pet Posts, handling Incoming Requests, reporting capabilities, Admin Management, and importantly, a Personality Test feature with Pet Suggestions to aid adoption. We'll now demonstrate some of these."

(2:00 - 12:00) Live Demo - Speaker 2

- *Speaker 2:* "Thanks [Speaker 1]. I'll now walk you through a live demo of FetchMeHome."
- **(Transition to Screen Share - Start at Login/Register Page)**
    - **1. Registration & Login (~1.5 min):** "For new users, registration is straightforward [Show Register form, fill quickly, submit]. Once registered, users can log in using their email and password [Show Login form, log in successfully]. Authentication is handled using JWT for secure sessions." [Briefly show logged-in home page/dashboard].
    - **2. Owner Posts Lost Pet (~3 min):** "Let's say I'm an owner and my pet is missing. I can navigate to the 'Services' or 'My Panel' area [Navigate] and choose 'Post Lost Pet'. I fill in the required details like name, age, type, description [Fill quickly], upload a photo [Select image], and use the integrated map feature to pinpoint the last known location [Briefly interact with map if possible]. Upon submission [Click Submit], this listing becomes available on the 'Find' page for the community to see."
    - **3. Finder Reports Found Pet (~3 min):** "Now, imagine another user is browsing the 'Find' page [Navigate to Find page] and spots a pet that matches our lost pet listing [Click on the listing]. They can read the details and if they believe they've found the pet, they click 'Verify Pet'. This prompts them to upload their own photo as evidence and provide their contact details [Select image, fill contact info]. Submitting this [Click Submit] sends a notification or request directly to the original owner for verification, facilitating the reunion."
    - **4. Personality Test & Suggestions ('Wow' Factor) (~3.5 min):** "A unique feature we've integrated is the Pet Personality Test [Navigate to Personality page]. Users answer questions about their lifestyle [Click through a few answers]. Once completed [Click Submit/See Results], the answers are processed by our backend, which communicates with the Ollama AI via our Singleton service to generate key personality traits [Point out traits]. Using these traits, we then query external Dog and Cat APIs to suggest suitable breeds that match the user's personality [Show suggested breeds]. Users can view details and even save pets they're interested in."
- *Speaker 2:* "That covers the main flows for posting, finding, and discovering pets on FetchMeHome. I'll hand over to [Speaker 3] to discuss the technical aspects."
- **(End Screen Share)**

(12:00 - 17:00) Technical Details - Speaker 3

- *Speaker 3:* "Thank you, [Speaker 2]."
- **(Slide 12: Tech Stack)**
    - "FetchMeHome is built entirely on the MERN stack: MongoDB for our NoSQL database, Express.js and Node.js powering our backend API, and React.js with CSS for the frontend user interface."

- **(Slide 14: Good SWE Practices)**
    - "Throughout the project, we emphasized good software engineering practices."

- **(Slide 15: Documentation - README)**
    - "We maintained a comprehensive README file serving as an introduction and providing clear installation procedures for future developers."
- **(Slide 16: Documentation - Comments)**
    - "Code clarity was enhanced through consistent commenting, improving readability and collaboration, as seen here in snippets of our backend logic."
- **(Slide 17: Code Reusability)**
    - "We focused on reusability. For instance, frontend elements like the AdminNavBar and Footer are built as separate components and reused across multiple pages, reducing code duplication."

- **(Slide 19: System Design)**
    - "Now, let's look at our system design."

- **(Slide 20: Architecture Diagram)**
    - "We utilized a layered architecture for our backend. User interactions on the UI trigger requests handled by Controllers, which process logic using data from Models, interacting finally with the Database."

- **(Slide 22: MVC Structure)**
    - "This structure closely follows the Model-View-Controller pattern, effectively separating the frontend View, backend Controller logic, and data Models."

- **(Slide 23: Design Patterns)**
    - "Beyond the overall architecture, we applied several specific design patterns:"

- **(Slide 24 & 25: MVC)**
    - "Firstly, the MVC pattern itself helped us organize the backend by separating data management (Models), request handling (Controllers), and API definitions

(Routes), leading to better maintainability and testability." *(Ensure Slide 24 Benefit text is corrected)*

- **(Slide 26 & 27: Middleware)**
    - "Secondly, we used the Middleware pattern in Express. Our authMiddleware, shown here, intercepts requests to handle JWT authentication cleanly, decoupling this security concern from our main controller functions." *(Ensure Slide 26 Benefit text is corrected)*

- **(Slide 28 & 29: Singleton)**
    - "Thirdly, for managing the connection to the external Ollama AI service, we implemented the Singleton pattern. By creating and exporting a single ollamaChat instance, we ensure efficient and consistent interaction with the AI across the backend."

- **(Slide 30: SOLID Principles)**
    - "We also considered SOLID principles. Key strengths include Single Responsibility in our Models, Interface Segregation through module imports and specific React props, and applying Liskov Substitution principles via composition with our frontend viewer components."

(17:00 - 19:00) Traceability Example - Speaker 4

- *Speaker 4:* "Thanks, [Speaker 3]. To demonstrate traceability from requirements to testing, let's focus on the 'Create Lost Pet Listing' use case."
- **(Slide 32: Use-case Description)**
    - "The requirement, Use Case #2-6 from our SRS, specifies that an authenticated owner must be able to create a listing for their missing pet."

- **(Slide 33: Class Diagram - *Minimize Time Here*)**
    - "This early class diagram shows some initial concepts..." *(Gesture generally)* "...but the flow is clearer in the sequence diagram."

- **(Slide 34: Sequence Diagram - *Use Correct Names Verbally*)**

    - "This sequence diagram shows the interaction: The user interacts via the **Frontend UI**, which triggers the **LostPetController** on the backend. The controller then uses the **LostPet Model** to create the record in the database."

- **(Slide 35: Good Designs Applied - Code Snippets)**
    - "In our code, this is implemented via the LostPetRoute which uses multer middleware and maps to the postLostPetRequest function in the LostPetController. This controller function then calls LostPet.create, interacting with the LostPetModel schema definition."

- **(Slide 36 & 37: Tests)**
    - "We verified this specific controller function using white-box testing. We analyzed its Control Flow Graph, derived 6 basis paths covering different scenarios, and executed test cases documented in our Lab 4 deliverables, confirming the logic operates correctly."

(19:00 - 20:00) Conclusion & Q&A - Speaker 1 or 4

- **(Slide 39: Thank You)**
    - "In summary, FetchMeHome utilizes the MERN stack and key design patterns like MVC, Middleware, and Singleton to provide a functional and maintainable platform for pet adoption and recovery. Thank you for your attention. We are now happy to answer any questions."