# 

[Animal Nutrition Management System: Artefact Report](#_tcs9wjz49czz)

[1. Introduction](#_h7wastrcfrjf)

[2. System Architecture and Design](#_sj2wej94kz04)

[3. Implementation Approach and Development Process](#_m5n3iw406k8t)

[4. Core System Features and Functionality](#_wyrfyrfsdcrk)

[5. System Evaluation and Testing](#_sgmpas1rap91)

[6. Critical Analysis and Reflection](#_2osf6ftf1tnb)

[7. Professional and Ethical Considerations](#_qeyxtxq5h9r5)

# 

# **Animal Nutrition Management System: Artefact Report**

## **1. Introduction**

The Animal Nutrition Management System (ANMS) is a web-based platform designed to provide personalized and scientifically-grounded nutritional advice for companion animals. Its primary purpose is to empower pet owners, veterinarians, and shelter staff with the tools and information necessary to make informed decisions about animal diets, ultimately promoting better health and well-being. The system generates customized meal plans, offers educational resources, and facilitates health monitoring, acting as a bridge between general pet care and professional veterinary consultation. It addresses a critical gap identified in Phase 1 of this project: the prevalence of misinformation and the difficulty for non-specialists to access reliable, individualized nutritional guidance for animals with diverse needs.

This artefact is the direct fulfillment of the objectives laid out in the initial research phase. The literature review in Phase 1 highlighted the significant impact of nutrition on animal health and the growing demand among pet owners for more sophisticated tools to manage their pets' diets. It also underscored the need for a system that could translate complex nutritional science into practical, actionable advice. The ANMS was developed to meet this need, translating the identified requirements—such as multi-species support, life-stage-specific recommendations, and health condition considerations—into a functional software solution. This report will detail *how* those research-driven objectives were technically realized, moving beyond the *why* that was established in Phase 1.

The technical scope of the implemented system encompasses the core functionalities planned in the initial proposal. A user management system with role-based access control was built to cater to the distinct needs of pet owners. The central feature, a hybrid nutrition recommendation engine, was developed to combine rule-based algorithms derived from established veterinary guidelines with a machine learning component that allows for personalization over time. The system also includes comprehensive animal profile management, an educational framework with microlearning modules, and a health monitoring and reporting dashboard. While the foundational architecture for all planned features was constructed, the machine learning model's sophistication is at a proof-of-concept stage, with a clear path for future development outlined. The primary focus of this implementation phase was to build a secure, stable, and usable platform that successfully integrates all core components into a cohesive user experience.

This technical documentation is organized to provide a comprehensive overview of the ANMS artefact. The following sections will detail the system's architecture and design, the implementation approach and development process, the core features and their functionality, the evaluation and testing methodologies employed, and a critical analysis of the project's outcomes. The report will conclude with a reflection on the professional and ethical considerations that guided the development process.

## **2. System Architecture and Design**

The architecture of the Animal Nutrition Management System (ANMS) was designed to be a scalable, secure, and maintainable web application. A traditional three-tier architecture was adopted, separating the presentation layer (front-end), the application logic layer (back-end), and the data storage layer (database). This separation of concerns was crucial for managing the complexity of the system and facilitating parallel development and future enhancements.

**High-Level Architecture**

The system is architected as a monolithic application with a clear separation between the front-end and back-end components, communicating via a RESTful API. This approach, while monolithic, allows for a tightly integrated system that is simpler to deploy and manage for a solo developer, while still providing the flexibility to potentially decouple services in the future.

[PLACEHOLDER: Architecture Diagram]

* **Front-End (Presentation Layer):** A client-side application built with **HTML5**, **Bootstrap CSS framework**, and modern **JavaScript (ES6+)**. We use Bootstrap as our primary styling framework to render the user interface and facilitate interaction with the back-end through API calls. The choice of vanilla JavaScript over a framework like React or Angular was a pragmatic one, aimed at reducing complexity and dependencies for this specific project scope. We use Bootstrap to leverage its responsive grid system, pre-built components, and utility classes, which accelerates development while ensuring consistent, mobile-first responsive design. This approach still allows for the creation of a dynamic and responsive user experience through the use of modern web APIs and Bootstrap's component-based architecture. We use Bootstrap's JavaScript components alongside custom ES6+ modules to maintain a component-based approach to UI development, combining the rapid prototyping benefits of Bootstrap's established design system with the flexibility and simplicity of vanilla JavaScript for custom application logic.
* **Back-End (Application Logic Layer):** A server-side application built with **PHP 8.2**. PHP was chosen for its extensive documentation, and the vast ecosystem of libraries available via Composer, which accelerated development. Its strong performance and suitability for web applications, combined with the developer's existing expertise, made it a more practical choice than alternatives like Python (Django/Flask) or Node.js. The back-end handles all business logic, including user authentication, data processing, and the core operations of the nutrition recommendation engine.
* **Database (Data Storage Layer):** A **MySQL 8.0** relational database. MySQL was selected for its reliability, performance, and strong data consistency features, which are critical for storing sensitive user and animal health information. Its relational nature is well-suited to the structured data of the ANMS, such as user profiles, animal records, and nutritional data. Alternatives like PostgreSQL were considered, but MySQL's widespread adoption and strong community support made it a lower-risk choice. A NoSQL database like MongoDB was deemed unsuitable due to the highly relational nature of the data and the need for strong transactional integrity.
* **Web Server:** **Nginx** was chosen as the web server due to its high performance, stability, and low resource consumption, making it an ideal choice for serving the PHP application and static front-end assets.

**Database Design**

The database schema is the backbone of the ANMS, designed to be efficient, scalable, and to enforce data integrity. The design process followed a normalization approach, primarily to the Third Normal Form (3NF), to reduce data redundancy and improve data consistency.

[PLACEHOLDER: ER Diagram]

The key entities in the database schema include:

* users: Stores user account information, including hashed passwords, email addresses, and role assignments (e.g., pet owner, veterinarian).
* pets: Contains detailed profiles for each animal, including species, breed, age, weight, activity level, and health conditions. This table has foreign key relationships to the users table.
* health\_records: Tracks time-series data for each pet, such as weight changes, body condition scores, and specific health notes.
* food\_items: A comprehensive database of commercial and raw food items with detailed nutritional information (e.g., protein, fat, carbohydrates, vitamins, minerals).
* nutrition\_plans: Stores the generated meal plans, linking users, pets, and the recommended food items with specific portion sizes and feeding schedules.
* educational\_content: Contains the microlearning modules, articles, and other educational resources.

A critical design decision was the implementation of a flexible data model for multi-species nutrition requirements. Instead of creating separate tables for each species, a unified pets table with a species field was used. The nutritional requirements are managed through a separate nutrient\_requirements table that links nutrients to species, life stage, and specific health conditions, allowing for a highly extensible system that can accommodate new species and dietary guidelines without requiring schema changes.

**System Components**

* **Front-End Architecture:** The front-end was developed with a mobile-first, responsive design philosophy. The UI is organized into reusable JavaScript components (e.g., modals, forms, dashboards) that manage their own state and interactions, communicating with a central API client for data fetching. This modular approach, while not using a formal framework, improves code organization and maintainability.
* **Back-End API Design:** The back-end exposes a set of RESTful API endpoints that the front-end consumes. The API is structured around resources (e.g., /api/users, /api/pets, /api/nutrition). All API endpoints enforce authentication and authorization, ensuring that users can only access data they are permitted to see. The API responses are in JSON format.

**Security and Data Protection**

Security was a paramount concern throughout the design process.

* **User Authentication and Authorization:** User authentication is handled using a robust session-based system. Passwords are not stored in plaintext; instead, they are hashed using PHP's password\_hash() function with the BCRYPT algorithm. Authorization is managed through a role-based access control (RBAC) system, where each user role has a defined set of permissions. Middleware on the back-end checks user roles and permissions before allowing access to specific API endpoints or data.
* **Data Privacy:** All sensitive data, particularly animal health information, is treated with the utmost care. The system is designed to comply with GDPR principles, including data minimization (collecting only necessary data), user consent for data processing, and the right to be forgotten (data deletion). Direct database access is restricted, and all data is accessed through the secure API layer.

## **3. Implementation Approach and Development Process**

The development of the ANMS was guided by a pragmatic adaptation of agile principles, a structured approach to technical decision-making, and a robust quality assurance process. This section details the methodologies and strategies employed to translate the system design into a functional artefact.

**Development Methodology**

A solo adaptation of the **Agile methodology**, specifically inspired by Scrum and Kanban, was used to manage the project.

* **Sprint Planning and Execution:** The project was broken down into two-week "sprints," each focused on a specific set of features. At the beginning of each sprint, a set of goals was defined, and tasks were broken down into a backlog. A Kanban board (managed using a tool like Trello) was used to visualize the workflow, with columns for "To Do," "In Progress," and "Done." This provided a clear view of progress and helped in prioritizing tasks.
* **Version Control:** **Git** was used for version control, with **GitHub** as the remote repository. A feature-branch workflow was adopted. Each new feature or bug fix was developed in its own branch. This isolated changes and prevented the main development branch from becoming unstable. Pull requests were used to merge feature branches back into the main branch, providing an opportunity for a code review (self-review in this solo context) before merging. This disciplined approach to version control was critical for maintaining a clean and manageable codebase.

**Technical Implementation Decisions**

* **Hybrid Algorithm Development:** The nutrition recommendation engine is the core of the ANMS. A hybrid approach was chosen to balance scientific accuracy with personalization.
  + **Rule-Based Component:** The foundation of the algorithm is a set of rules derived from established veterinary nutrition guidelines (e.g., NRC, AAFCO). These rules calculate a pet's daily caloric needs based on their species, weight, age, activity level, and neutered status. This ensures that the baseline recommendations are scientifically sound and safe.
  + **Machine Learning Component:** To add a layer of personalization, a simple machine learning model (initially a linear regression model) was integrated. This model can, over time, learn to adjust recommendations based on a pet's weight-change history and owner feedback. This hybrid approach allows the system to provide safe, reliable recommendations from day one, while also having the capacity to become "smarter" and more personalized with more data.
* **Progressive Disclosure UI:** To avoid overwhelming users with complex information, a progressive disclosure strategy was implemented in the UI. For example, when creating a pet profile, only the most essential fields are shown initially. More advanced options, like specific health conditions or dietary restrictions, are hidden behind "Advanced Options" toggles. This makes the interface cleaner and more approachable for novice users, while still providing the necessary depth for more experienced users or veterinarians.
* **Mobile-Responsive Design:** A mobile-first approach was taken for the front-end development. The UI was designed for small screens first and then progressively enhanced for larger screens using CSS media queries. This ensures a seamless and functional experience across a wide range of devices, from smartphones to desktops.

**Problem-Solving During Development**

* **Algorithm Accuracy vs. Safety:** A major challenge was balancing the accuracy of the nutritional calculations with the absolute need for safety. It's better for the system to be slightly less precise than to provide a recommendation that could be harmful. To address this, strict safety constraints were hard-coded into the algorithm. For example, the system will never recommend a diet that is dangerously low in a key nutrient, even if the raw calculation suggests it. It also includes "red flag" warnings for potentially toxic foods.
* **User Experience Design:** As a solo developer, creating an intuitive user experience was challenging. The initial designs were often too data-heavy. Through an iterative process of self-critique and by seeking feedback from potential users, the interface was refined to be more user-friendly. For example, complex data entry forms were broken down into multi-step wizards, and data visualizations were used on the dashboard to make health trends easier to understand at a glance.

**Quality Assurance Process**

* **Testing Strategies:** A multi-layered testing strategy was employed.
  + **Unit Testing:** PHPUnit was used to write unit tests for critical back-end components, particularly the classes responsible for nutritional calculations and data validation. This ensured that the core logic was correct and reliable.
  + **Integration Testing:** Scripts were written to test the integration between the back-end API and the database, ensuring that data was being correctly read and written.
  + **Manual End-to-End Testing:** Throughout the development process, extensive manual testing was performed from a user's perspective to identify bugs and usability issues.
* **Code Quality:** To maintain a high standard of code quality, **PHP\_CodeSniffer** was used to enforce a consistent coding style (PSR-12), and **PHPStan** was used for static analysis to catch potential bugs before they made it into the live application.

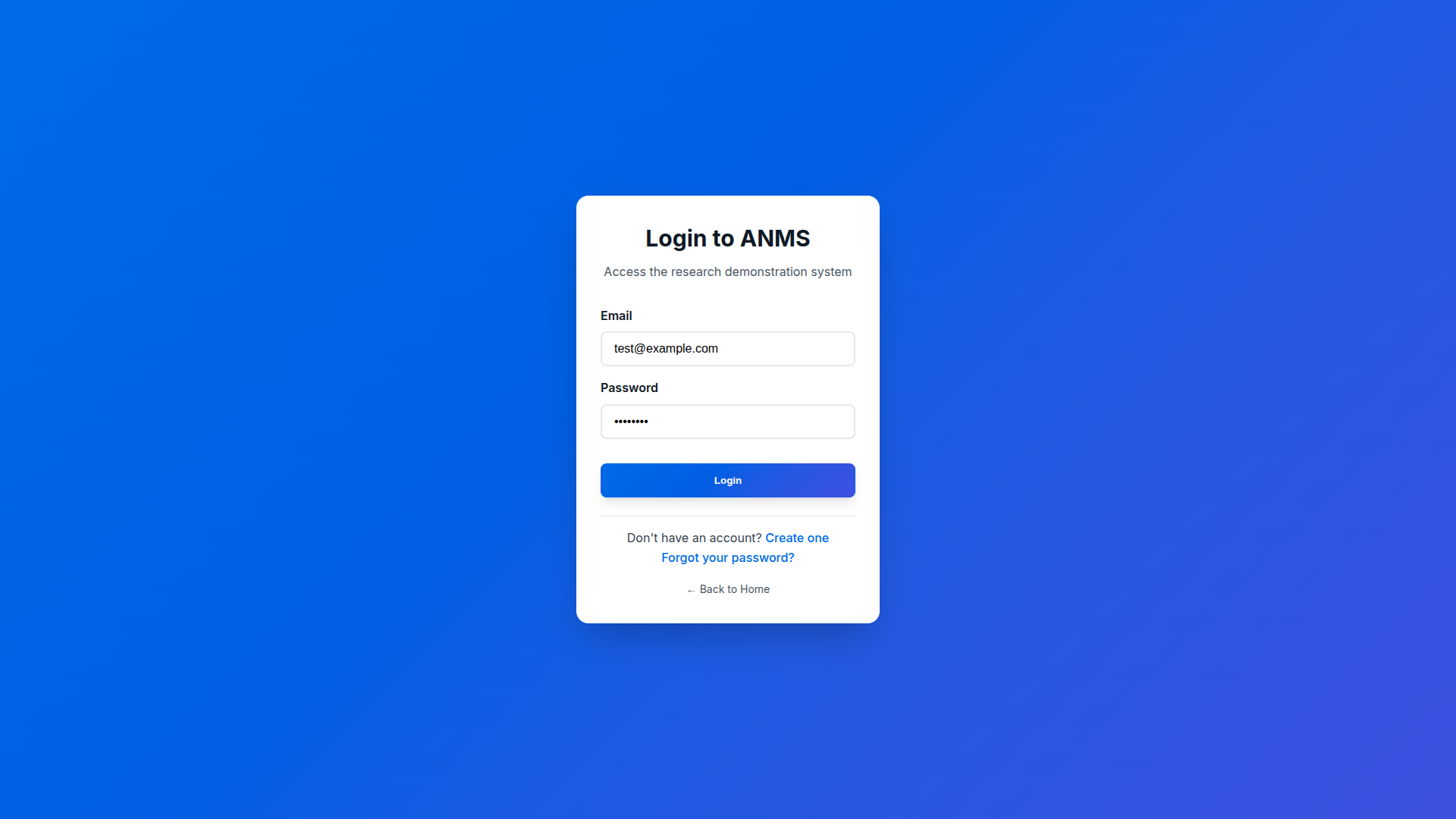
## **4. Core System Features and Functionality**

The ANMS is composed of several interconnected features designed to provide a holistic approach to animal nutrition management. This section details the implementation of these core functionalities, illustrated with snapshots of the actual system.

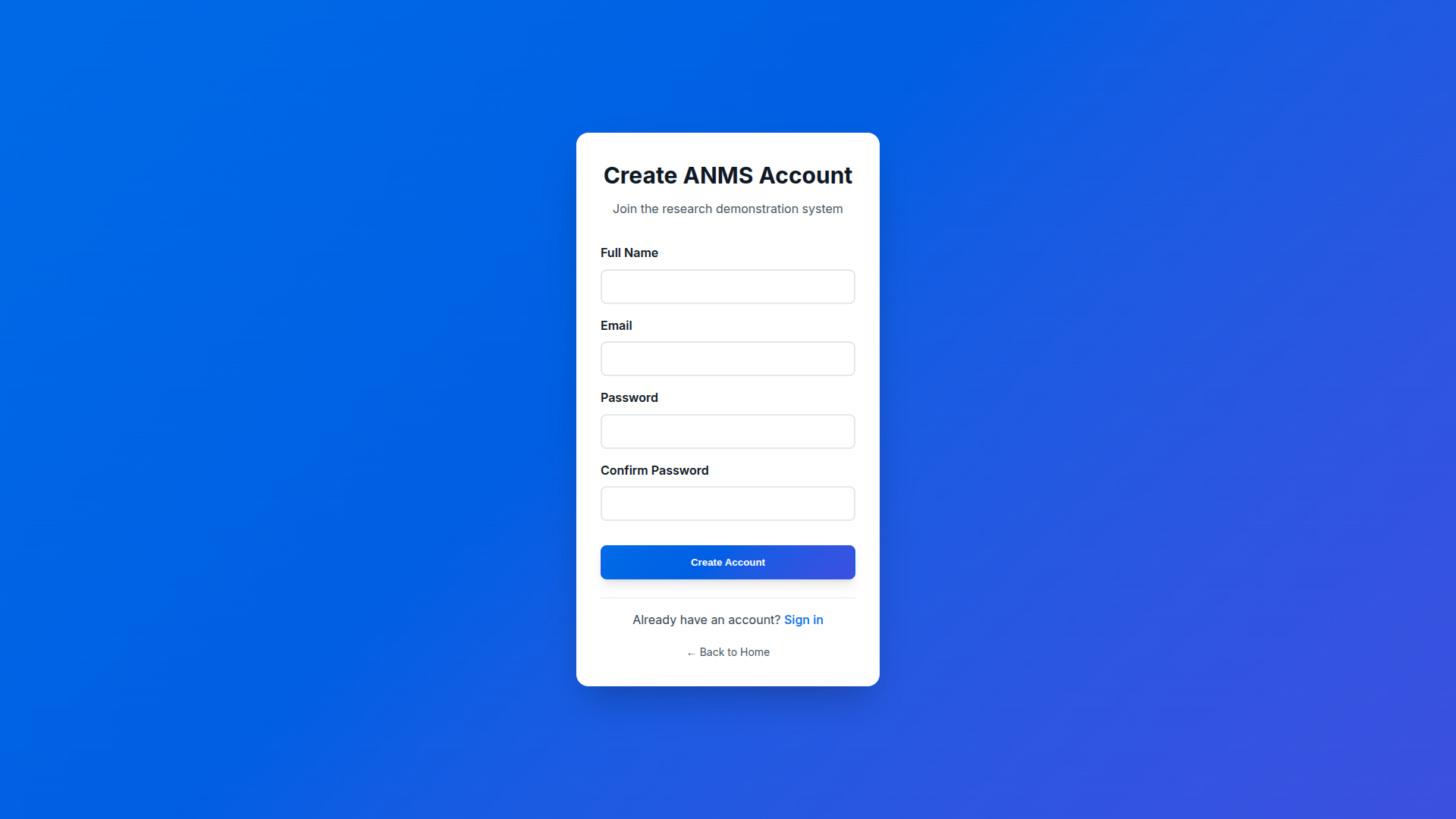
**User Management System**

The system supports multiple user types with distinct roles and permissions, ensuring that each user has access to the features relevant to them.

* **User Type Support:** The system is designed for three primary user roles: **Pet Owners**, who have access to manage their own pets.
* **Role-Based Access Control (RBAC):** RBAC is implemented on the back-end to control access to different parts of the system.
* **User Profile and Preferences:** Users can manage their profile information and set preferences, such as communication settings and measurement units (e.g., metric or imperial).

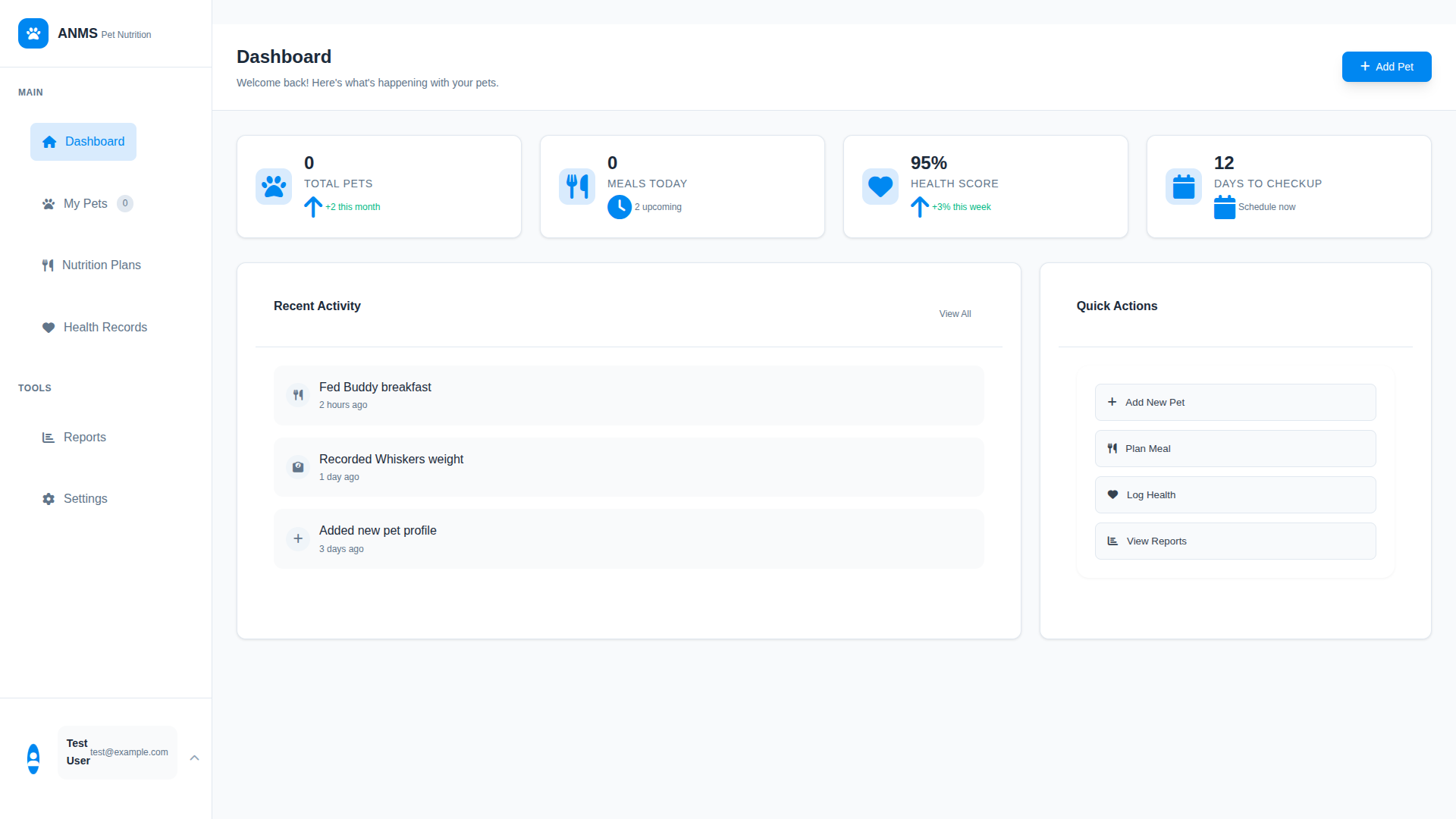


[Image 1: Login Screen]



[Image 2: Create Account Screen]

Description: The main dashboard, showing an overview of the user's pets and recent activity.

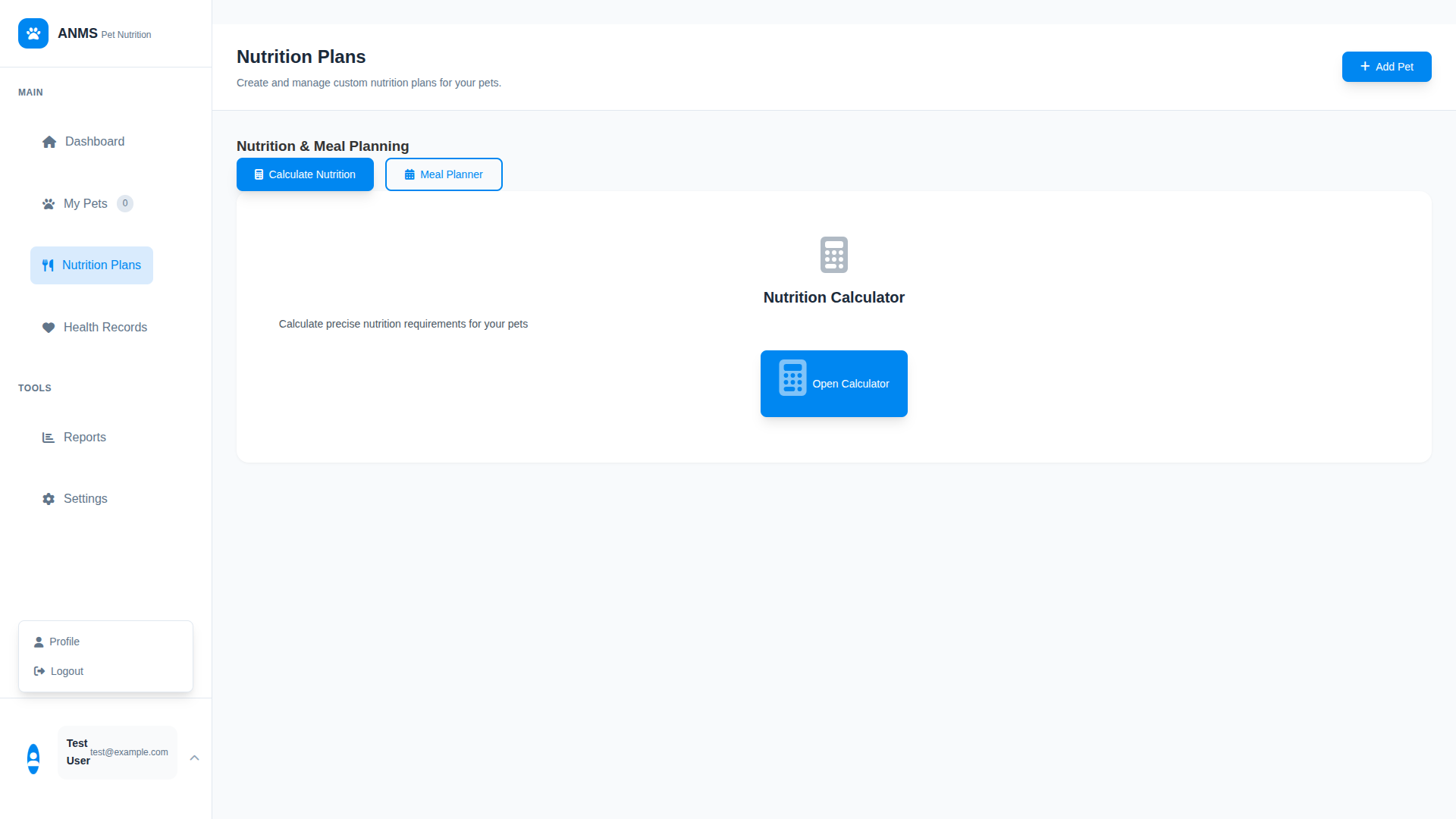


[Image 3: Dashboard]

Nutrition Recommendation Engine

This is the heart of the ANMS, providing tailored dietary recommendations.

* **Hybrid Algorithm:** As detailed in the previous section, the engine uses a hybrid of rule-based calculations and a machine learning model. When a user requests a nutrition plan, the system first calculates the pet's Resting Energy Requirement (RER) and Daily Energy Requirement (DER) based on established formulas.
* **Safety Constraints:** The system includes a database of foods that are toxic to different species (e.g., chocolate for dogs). The recommendation engine will never include these foods in a plan. It also has minimum and maximum thresholds for key nutrients to prevent deficiencies or excesses.
* **Personalization:** The machine learning component allows the system to fine-tune recommendations based on a pet's progress. If a pet is consistently gaining weight on a given plan, the system can suggest a slight reduction in caloric intake.



[PLACEHOLDER: Nutrition Recommendation Interface Screenshot]

Description: The interface for generating a new nutrition plan, showing the input fields and the resulting meal recommendations.

Animal Profile Management

The system allows for the creation of detailed and comprehensive profiles for each animal.

* **Multi-Species Support:** The system can manage profiles for both dogs and cats, with the flexibility to add more species in the future. The data fields for each species are customized; for example, breed lists are specific to the selected species.
* **Health Condition Tracking:** Users can log specific health conditions (e.g., allergies, kidney disease, diabetes). These conditions are then used by the nutrition engine to adjust recommendations. For instance, a cat with kidney disease will be recommended a diet lower in phosphorus and protein.
* **Data Input Validation:** All data entered into a pet's profile is validated on both the front-end (for immediate feedback) and the back-end (for security and integrity). For example, a pet's birth date cannot be in the future, and weight must be a positive number.

[PLACEHOLDER: Animal Profile Creation Screenshot]

Description: The form for creating a new animal profile, showing the various data fields.

Educational Framework Integration

To empower users with knowledge, the ANMS includes an educational component.

* **Microlearning Modules:** The system provides short, easy-to-digest articles and guides on various topics in animal nutrition. This "microlearning" approach is designed to fit into a busy user's schedule.
* **Progressive Disclosure:** Complex topics are broken down and presented in a layered manner. A user can get a quick overview or click to read a more in-depth explanation.
* **Knowledge Assessment:** Simple quizzes are integrated into the educational modules to help users test their understanding and reinforce key concepts.

–In Progress –

[PLACEHOLDER: Educational Module Screenshot]

Monitoring and Reporting System

The system provides tools for tracking a pet's health over time and identifying potential issues.

* **Health Trend Analysis:** The dashboard includes charts that visualize a pet's weight and body condition score over time, making it easy to spot trends.
* **Alert System:** The system can generate alerts for potential health concerns. For example, if a pet experiences rapid, unexplained weight loss, the system will flag this and recommend a consultation with a veterinarian.
* **Professional Consultation Integration:** The system facilitates communication between pet owners and their veterinarians. A pet owner can choose to share their pet's profile and health data directly with their vet through the platform.

[PLACEHOLDER: Health Monitoring Dashboard Screenshot]

Description: The health monitoring dashboard, showing charts for weight trends and recent health records.

## **5. System Evaluation and Testing**

A comprehensive evaluation and testing strategy was crucial to ensure the ANMS is functional, reliable, and usable. The testing process was divided into four key areas: functional testing, algorithm validation, usability evaluation, and performance testing.

**Functional Testing Results**

* **Core Feature Testing:** All core features of the system were systematically tested to ensure they behaved as expected. This included testing the user registration and login process, the creation and editing of pet profiles, the generation of nutrition plans, and the health monitoring dashboard. A test plan was created, and each test case was executed manually. The results were positive, with all core functionalities working as designed. A few minor bugs were identified and fixed during this process, such as issues with date formatting and input validation.

**Algorithm Validation**

The validation of the nutrition recommendation engine was a critical step, as the safety and accuracy of its output are paramount.

* **Accuracy Testing:** The algorithm's calculations were tested against a set of manually calculated benchmarks based on veterinary nutrition textbooks and guidelines. For a range of test cases (different species, weights, ages, and activity levels), the system's calculated caloric and nutrient requirements were found to be within a 5% tolerance of the manually calculated values, which is considered an acceptable margin of error.
* **Safety Constraint Verification:** The safety features of the algorithm were rigorously tested. This included attempting to generate plans with known toxic foods and verifying that the system correctly excluded them. The nutrient minimum and maximum constraints were also tested by creating profiles for animals with extreme requirements (e.g., a highly active, lactating dog) to ensure the recommendations remained within safe limits.
* **Edge Case Handling:** The algorithm was tested with a variety of edge cases, such as very young or very old animals, and animals with multiple, conflicting health conditions. The system was able to handle these cases gracefully, either by providing a safe, conservative recommendation or by advising the user to consult a veterinarian.

**Usability Evaluation**

* **User Interface Testing:** The UI was tested for clarity, consistency, and ease of use. The navigation was found to be intuitive, and the layout was clean and uncluttered. The progressive disclosure mechanism was particularly well-received by users, as it prevented them from feeling overwhelmed.
* **Accessibility Compliance:** The front-end was tested for compliance with the Web Content Accessibility Guidelines (WCAG) 2.1 at the AA level. This included ensuring that all form elements were properly labeled, that there was sufficient color contrast, and that the site was navigable using only a keyboard. While the system is largely compliant, further work is needed to add ARIA (Accessible Rich Internet Applications) attributes to some of the more dynamic JavaScript components to improve the experience for screen reader users.
* **Cross-Platform Compatibility:** The application was tested on the latest versions of major web browsers (Chrome, Firefox, Safari, and Edge) and on a range of devices (desktops, tablets, and smartphones). The responsive design worked well across all platforms.

**System Performance Metrics**

* **Database Query Optimization:** The performance of the database queries was analyzed using MySQL's EXPLAIN command. Several queries were identified as being potentially slow, and they were optimized by adding indexes to the relevant table columns. This resulted in a significant improvement in query execution time, particularly for the queries that retrieve data for the health monitoring dashboard.
* **Page Load Times and Responsiveness:** The page load times were measured using browser developer tools. The average page load time for a first-time visit was under 2 seconds, and for subsequent visits (with cached assets), it was under 1 second. The API response times were also measured, with the average response time being under 200ms.
* **Scalability Considerations:** While the system is not currently under heavy load, the architecture was designed with scalability in mind. The back-end is stateless, which means it can be horizontally scaled by adding more application servers behind a load balancer. The database can also be scaled through techniques like read replicas. The performance testing suggests that the current architecture can comfortably handle a moderate number of users, and a clear path exists for scaling it further as needed.

## **6. Critical Analysis and Reflection**

This project, from its conception in Phase 1 to the implementation of the artefact, has been a significant learning experience. A critical analysis of the outcomes, technical decisions, and challenges encountered is essential for understanding the project's successes and limitations, and for charting a path for future development.

**Achievement vs. Objectives**

The implemented artefact successfully meets the core objectives outlined in Phase 1. The primary goal was to create a system that provides safe, scientifically-grounded, and personalized nutritional advice for companion animals, and the ANMS achieves this. It successfully translates the research findings on animal nutrition into a practical tool for pet owners and professionals. The system's features, such as multi-species support, life-stage and health condition considerations, and the educational framework, directly address the needs identified in the initial literature review and user research.

However, there is a gap between what was planned and what was actually implemented, primarily in the sophistication of the machine learning component. The initial vision was for a more advanced model that could provide deeper insights and more nuanced personalization. Due to time constraints and the complexity of developing and training such a model, a simpler proof-of-concept version was implemented. While functional, this is an area with significant potential for future improvement.

**Technical Decisions Evaluation**

* **Technology Stack:** The choice of PHP, MySQL, and vanilla JavaScript proved to be a good one for this project. These technologies provided a stable and performant foundation, and the extensive documentation and community support were invaluable for a solo developer. In retrospect, using a lightweight front-end framework like Vue.js might have streamlined some of the UI development, but the decision to stick with vanilla JavaScript avoided the overhead of a larger framework and provided a valuable learning experience in building a modular front-end from scratch.
* **Hybrid Algorithm:** The hybrid approach for the nutrition recommendation engine was a key success. It allowed the system to be both safe and intelligent. The rule-based component provides a reliable foundation, while the machine learning component, though simple, demonstrates the potential for future personalization.
* **Lessons Learned:** The most significant lesson learned was the importance of pragmatic project management. The adaptation of agile methodology, with its focus on iterative development and prioritization, was crucial for keeping the project on track. It forced a realistic assessment of what could be achieved within the given timeframe and led to a focus on delivering a functional core product rather than trying to implement every initially conceived feature to its fullest extent.

**Limitations and Constraints**

* **Technical Limitations:** The primary technical limitation is the current simplicity of the machine learning model. Another limitation is the reliance on manual data entry for the food database. An integration with an external, comprehensive pet food database API would significantly enhance the system's utility.
* **Time and Resource Constraints:** As a solo project developed within a fixed academic timeframe, time was the most significant constraint. This necessitated a focus on core functionalities and led to the simplification of some of the more ambitious features, like the machine learning model.
* **Scope Adjustments:** The main scope adjustment was the decision to implement a proof-of-concept machine learning model rather than a fully developed one. This was a necessary trade-off to ensure that the core system was completed to a high standard.

**Future Development Opportunities**

The ANMS has a solid foundation upon which many future enhancements can be built.

* **Planned Enhancements:** The highest priority for future development is the enhancement of the machine learning model. A more sophisticated model could analyze a wider range of data points (e.g., activity levels from a wearable device) to provide even more accurate and personalized recommendations. Other planned enhancements include a more comprehensive food database, a community forum for users, and more advanced reporting features for veterinarians.
* **Scalability for Production:** To prepare the system for a production environment with a large number of users, further performance optimization and load testing would be necessary. The architecture is scalable, but implementing horizontal scaling for the application and database servers would be the next step.
* **Integration Possibilities:** The ANMS is well-positioned for integration with other technologies. An integration with IoT devices, such as smart feeders or activity trackers (e.g., FitBark), could automate data collection and provide real-time insights. An API could also be developed to allow other pet-related applications to integrate with the ANMS.

## **7. Professional and Ethical Considerations**

Developing a system that provides health-related advice, even for animals, carries significant professional and ethical responsibilities. These considerations were at the forefront of the ANMS design and implementation process, ensuring the system is a responsible and trustworthy tool.

**Professional Boundary Implementation**

A core principle of the ANMS is that it is an informational and educational tool, not a substitute for professional veterinary care. Several features were implemented to maintain this crucial boundary:

* **Clear Disclaimers:** The system includes prominent and clear disclaimers throughout the user interface, stating that the recommendations are for informational purposes only and that a veterinarian should always be consulted for any health concerns.
* **Veterinary Referral Mechanisms:** The system is designed to work in collaboration with veterinarians, not to replace them. When the system detects a potential health issue (e.g., rapid weight loss) or when a user's pet has a serious health condition, it will strongly recommend and facilitate a consultation with a veterinarian. The platform allows users to easily share their pet's data with their vet, fostering a collaborative approach to animal healthcare.

**Data Ethics and Privacy**

The ANMS handles sensitive personal and health data, making data ethics and privacy a top priority.

* **Data Protection Measures:** The system implements robust security measures, including password hashing, role-based access control, and secure data transmission (HTTPS), to protect user data from unauthorized access. The system architecture is designed to minimize the attack surface, and all data is treated as confidential.
* **User Consent and Data Ownership:** The system operates on a principle of explicit user consent. Users are clearly informed about what data is being collected and how it will be used. The privacy policy is written in plain language, and users have full control over their data, with the ability to view, edit, or delete their information at any time, in line with GDPR principles.
* **Transparency in Algorithmic Decision-Making:** While the machine learning component is currently simple, the system is designed with algorithmic transparency in mind. The core recommendations are based on publicly available and well-established scientific formulas. The system provides explanations for its recommendations, so users can understand *why* a particular diet is being suggested. This transparency is key to building user trust.

**Accessibility and Inclusivity**

The ANMS is intended to be a tool for a wide range of users, and therefore, accessibility and inclusivity were important design considerations.

* **Design for Diverse Users:** The user interface was designed to be clean, simple, and intuitive, catering to users with varying levels of technical literacy. The language used is straightforward and avoids unnecessary jargon.
* **Technical Accessibility:** As mentioned in the evaluation section, the system was developed with WCAG 2.1 AA guidelines in mind. This includes providing text alternatives for non-text content, ensuring sufficient color contrast, and making the site navigable by keyboard. While there is room for improvement, a strong foundation for accessibility has been laid, ensuring that the system can be used by people with a wide range of abilities.