



university of
 groningen

faculty of science
 and engineering

Design and Implementation of an AI-Research Management Platform for the Healthcare Domain

Bachelor's Project Computing Science

July 2024

Author: Oscar de Francesca

Student Number: S4773012

First supervisor: Prof. Dimka Karastoyanova

Second supervisor: Michel Medema, MSc.

Abstract

This thesis presents the development of the LLACE¹ platform, an AI-Research Management Platform specifically designed for the healthcare domain. The platform addresses the need for a comprehensive system to manage the lifecycle of AI models in medical research, which includes data integration, model evaluation, and deployment. The platform is particularly tailored to the requirements of the University Medical Center Groningen (UMCG), emphasizing secure data management and compliance with healthcare standards.

The methodology involved an extensive review of existing AI and healthcare informatics literature and detailed user requirements analysis through interviews with UMCG researchers. The LLACE platform integrates advanced search capabilities, user and project management, and robust data handling features. It also facilitates the deployment of AI models.

The LLACE platform's modular and extensible architecture ensures scalability and maintainability, supporting the seamless collaboration of researchers. The platform's design promotes transparency and reproducibility in AI-driven healthcare innovations, making it a significant contribution to the technological infrastructure of medical research. This project aims to enhance the efficiency and effectiveness of healthcare services by providing a robust tool for managing AI research activities.

¹Named after Ada Lovelace, a pioneer in computer science.

Contents

1	Introduction and Motivation	6
1.1	Proposal	7
2	Related Work	8
2.1	Research Management Systems	8
2.1.1	Overview of Existing Platforms	8
2.1.2	Limitations and Challenges	8
2.2	Model Management Systems	9
2.2.1	Overview of Existing Platforms	10
2.2.2	Limitations and Challenges	10
2.3	Contribution of LLACE	10
2.4	Search Process	11
3	Requirements	12
3.1	Requirements Gathering	12
3.2	System Functionalities	12
3.3	MoSCoW Technique	13
3.4	Functional Requirements	14
3.5	Non-Functional Requirements	17
4	Architecture	19
4.1	Ideal Architecture	19

4.1.1	System Context Diagram	19
4.1.2	Container Diagram	21
4.1.3	Backend Component Diagram	22
4.1.4	Ideal High-Level Diagram	24
4.2	MVP Architecture	27
4.2.1	MVP High-Level Diagram	27
5	Design	28
5.1	Code Design	29
5.2	Frontend Design	29
5.3	Modularity	30
5.4	Extensibility Decisions	30
6	Methods	31
6.1	Data Selection	31
6.2	Technology Stack	31
6.2.1	Ideal Technology Stack	31
6.2.2	MVP Technology Stack	33
6.3	Overview of UbiOps	34
6.3.1	Introduction to UbiOps	34
6.3.2	Why UbiOps	34
6.3.3	Functionalities of UbiOps	35
6.3.4	Alternatives	35
7	Evaluation	37
7.1	Functional Requirements	37

7.2	Non-Functional Requirements	38
8	Conclusion	40
9	Future Work	41
9.1	Model Deployment and Performance Metrics	41
9.2	Automated Tag Generation	41
9.3	Advanced Search Capabilities	41
9.4	Enhanced Information for Projects and Models	42
9.5	User Feedback and Iterative Improvements	42
9.6	Integration with Additional Data Sources	42
9.7	Scalability and Performance Optimization	42
9.8	Security and Compliance Enhancements	43
10	Acknowledgments	44
A	Meeting Notes: Researcher Requirements Gathering - Anonymized	48
B	LLACE Frontend Design Illustrations	50

List of Figures

1	System Context Diagram	20
2	Container Diagram	21
3	Backend Component Diagram	23
4	Ideal High-Level Diagram	25
5	MVP High-Level Diagram	28
6	Account Page: Users can view and edit their profile information. .	50

7	Dashboard: Users can access various functionalities such as managing and creating Research Projects and AI models.	51
8	Edit Project Page: Users can modify details of their research projects.	52
9	General Search: Provides powerful search capabilities across projects, models, profiles, and publications.	53
10	Model Creation: Users can upload and configure new AI models.	54
11	Model Deployment: Facilitates deploying models to UbiOps.	54
12	Project Creation: Users can create new research projects.	55
13	Project Page: Provides detailed information about a specific project, including title, description, additional details, collaborators, and related models, projects, and publications.	56

List of Tables

1	Color coding schema for MoSCoW Requirements based on interviews and literature analysis (see Appendix A).	14
2	Functional Requirements based on interviews and literature analysis (see Appendix A).	15
3	Functional Requirements (continuation of Table 2), based on interviews and literature analysis (see Appendix A).	16
4	Non-Functional Requirements based on interviews and literature analysis (see Appendix A).	18
5	Ideal Technology Stack for LLACE	32
6	MVP Technology Stack for LLACE	34
7	Evaluation of Functional Requirements	37
8	Evaluation of Functional Requirements (continuation of Table 7)	38
9	Evaluation of Non-Functional Requirements	39

1 Introduction and Motivation

The integration of Artificial Intelligence (AI) in healthcare represents a transformative opportunity, promising significant advancements in diagnosis, treatment, and operational efficiency. However, the deployment and management of AI-driven solutions in this field are fraught with challenges, including data privacy concerns, the need for high accuracy, and the integration of diverse data sources [40, 19]. These challenges underscore the necessity for a specialized platform to manage the lifecycle of AI model development and maintenance in healthcare settings.

This thesis proposes the design and implementation of an AI Research Management Platform, LLACE, specifically tailored for the healthcare domain. The platform addresses the critical need for a system that not only manages the technical aspects of AI model research, development, and deployment but also aligns with the ethical and operational standards required in healthcare research [40, 19]. Unlike general-purpose AI platforms, LLACE is designed to cater to the unique requirements of medical research, such as stringent data security, regulatory compliance, and the need for accurate and reproducible results [4, 11].

The healthcare sector, particularly institutions like the University Medical Center Groningen (UMCG), faces distinct challenges in AI research management. Currently, researchers at UMCG often work in isolation, with research data and models stored locally or on disparate systems. This fragmentation hinders collaboration, slows the translation of research into clinical practice, and poses risks to data security and integrity [4]. UMCG researchers currently utilize the Habrok HPC cluster for intensive computational tasks [24]. However, they lack a convenient way to collaborate and share their research, underscoring the need for a centralized platform.

The proposed platform, LLACE, aims to streamline the research process, enhance collaboration among scientists, and expedite the deployment of AI innovations in healthcare. It seeks to create an ecosystem that supports the entire lifecycle of AI models—from development and validation to deployment and monitoring. By centralizing these processes, LLACE addresses several critical gaps identified in current systems, such as the lack of integrated model repositories, inadequate support for multi-disciplinary collaboration, and insufficient mechanisms for ensuring model reproducibility and ethical compliance [38, 25].

The development of LLACE is driven by the overarching research question: “How can we design and implement an AI-research management platform that effectively caters to the unique needs and challenges of the healthcare domain?” This question encapsulates the thesis’s focus on creating a robust and adaptable platform that not only meets the technical requirements of AI research but also aligns with the unique needs and requirements of healthcare, an industry that has an extremely narrow tolerance for error and a critical need for broad innovation and improvement.

To ensure that the platform meets these goals, the research will be guided by several sub-

questions, including:

- How can the platform effectively integrate with existing systems to streamline data collection and ensure compatibility?
- What are the key user requirements and workflow considerations for healthcare researchers that must be taken into account in the platform's user interface and functionality design?
- What infrastructure requirements and considerations are necessary to support the scalability, reliability, and performance of the platform?
- How can the platform facilitate collaboration and knowledge sharing among multidisciplinary teams of researchers, clinicians, data scientists, and other stakeholders involved in healthcare research projects?

1.1 PROPOSAL

We propose to design a system that specifically caters to the needs of researchers working in the healthcare domain, particularly at UMCG. This system will enable researchers to efficiently discover, manage, and collaborate on AI research projects. The platform will address the unique challenges posed by the healthcare environment, including data security, regulatory compliance, and the need for high accuracy in AI applications. By leveraging existing literature and eliciting detailed requirements from UMCG researchers, the project aims to develop a comprehensive tool that aligns with their specific needs and integrates seamlessly with their current infrastructure. The system will include essential functionalities such as user management, data and profile storage, advanced search capabilities, and an AI model repository. The primary objective is to establish a robust and flexible architecture that supports the platform's scalability, security, and user accessibility.

The outcome of this thesis will be a detailed architectural design for LLACE, which will serve as a blueprint for future development and implementation. This architecture will not only meet the current needs of UMCG but will also be adaptable for potential future expansions and integrations with other systems.

2 Related Work

The rapid proliferation of AI research and machine learning (ML) models across various domains necessitates robust systems to manage the entire lifecycle of these models, from inception to retirement [38]. This section reviews the current landscape of research and model management systems, highlighting their strengths and limitations, and how the proposed LLACE platform aims to address these gaps.

2.1 RESEARCH MANAGEMENT SYSTEMS

Research Management Systems (RMS) are essential for organizing, managing, and tracking research activities. This section provides an overview of existing platforms and discusses their limitations and challenges.

2.1.1 OVERVIEW OF EXISTING PLATFORMS

Among the widely used open-source RMS platforms are DSpace and ePrints. DSpace is a repository software that facilitates the creation of open access repositories for scholarly and published digital content. Developed by MIT and HP, DSpace offers robust features for archiving and disseminating digital content, including advanced metadata management, interoperability, and comprehensive search capabilities [21].

Similarly, ePrints, developed by the University of Southampton, is another popular platform in academic settings. It provides a user-friendly interface for managing research outputs, with features such as customizable metadata, easy content submission workflows, and support for various data types [8].

2.1.2 LIMITATIONS AND CHALLENGES

Despite their strengths, both DSpace and ePrints have significant limitations, especially when it comes to creating a specialized platform like LLACE. A major issue with these platforms is the complexity of their search interfaces. For instance, DSpace users “often preferred the advanced search form due to its detailed filtering options,” but found the single-box search form inadequate as it frequently returned an overwhelming number of irrelevant results without effective refinement mechanisms [20].

In contrast, ePrints featured a simpler three-box search form, which users generally preferred for its simplicity. However, the advanced search form in ePrints was criticized for being overly complex, discouraging use among less experienced users. Both platforms were also criticized for inadequate user guidance during the search process, leading to user frustration

and inefficiencies [20]. For example, one user noted, “When you type ‘bibliometrics’ in the search box, only the results which have ‘bibliometrics’ in their title are returned. If full-text search is made possible, one could access any time to an article dated in 2012, which is what I’m looking for” [43]. This issue emphasizes the need for comprehensive search capabilities, including matching search queries with more than only result titles, which LLACE aims to provide.

Additionally, these platforms do not adequately meet the unique needs of healthcare AI research, such as managing sensitive data, ensuring regulatory compliance, and supporting the lifecycle management of AI models. This is crucial because the effective management of AI models, including versioning, tracking, and ensuring reproducibility, is vital in healthcare research but is not adequately addressed by DSpace or ePrints [43]. For example, a user commented, “Becoming a member is difficult” [43], highlighting user onboarding issues that LLACE will address by integrating with the UMCG Single Sign-On system for seamless access.

Another significant limitation of existing RMS platforms like DSpace and ePrints is their lack of support for managing data during the initial stages of research activities. According to an analysis conducted by Amorim et al., various research data management platforms, including DataFlow’s DataStage and DataBank, and the Dendro platform, highlight the necessity for integrated environments that manage and describe data interactively. These platforms, reviewed in the analysis, demonstrate the importance of flexibility and customization in data management, allowing researchers to add specific details, tags, and metadata to their projects [2]. LLACE aims to incorporate these capabilities, offering a high degree of customizability to enable users to manage and categorize data according to their specific needs, thus addressing the shortcomings of existing platforms.

While DSpace and ePrints provide solid foundations for academic research management, they are not designed to handle the specialized requirements of healthcare research. The platforms are also criticized for being overly complex and not user-friendly enough, which can hinder their usability. As such, they are not suitable as a base for LLACE. However, these platforms have set a strong foundation, and if more time and resources were available, it might be worth considering building upon them. For now, LLACE will focus on a simpler, custom RMS solution that incorporates the best features of these systems while avoiding their pitfalls.

2.2 MODEL MANAGEMENT SYSTEMS

Model Management Systems (MMS) are crucial for the efficient deployment, maintenance, and tracking of ML models. This section provides an overview of existing platforms and discusses their limitations and challenges.

2.2.1 OVERVIEW OF EXISTING PLATFORMS

Notable systems include ModelDB and MLFlow, which offer essential capabilities such as versioning, tracking, and sharing of ML models [36, 42]. These features are critical for managing the lifecycle of models, from development to deployment and beyond.

In the commercial sector, platforms like SAS Model Manager, Uber’s Michelangelo, and Facebook’s FBLearner Flow cater to specific organizational needs by providing tailored model management solutions that align with different operational requirements [31, 33, 9].

2.2.2 LIMITATIONS AND CHALLENGES

Despite advancements in model management, several challenges persist, including ensuring reproducibility, managing model versioning, and integrating models into production environments. Many existing systems lack the adaptability needed to accommodate diverse requirements across different domains, particularly in specialized fields like healthcare [38, 36, 42].

These systems also often fall short in addressing critical healthcare needs, such as regulatory compliance and secure data handling. While they provide robust technical tools, they do not always align with the stringent ethical and operational standards required in healthcare research.

2.3 CONTRIBUTION OF LLACE

The LLACE platform is designed to address the specific needs of healthcare research that existing RMS and MMS like DSpace, ePrints, ModelDB, and MLFlow do not adequately meet. While these platforms are effective in their respective domains, they are too large and complex for the specific needs of LLACE, where specialized data management is crucial. LLACE is designed as a more lightweight and custom solution, tailored specifically for the University Medical Center Groningen (UMCG) and similar institutions.

LLACE will incorporate a more intuitive search functionality, balancing simplicity and advanced filtering options to enhance user experience. It will feature a single-box search interface with dynamic filter options, allowing users to refine their searches effectively. Additionally, LLACE will provide comprehensive support for managing sensitive healthcare data, ensuring compliance with regulatory standards, and supporting the entire lifecycle of AI models, including aspects like versioning, tracking, and ensuring reproducibility.

Furthermore, LLACE emphasizes high customizability in data management, allowing users to add specific details, tags, and metadata to their research projects and AI models. This flexibility is crucial for accommodating the dynamic and complex nature of healthcare research data, making LLACE a more suitable choice than extending existing platforms like

DSpace or ePrints.

Additional features of LLACE will be detailed in later sections, particularly in the upcoming Requirements section.

2.4 SEARCH PROCESS

The literature search for this thesis was conducted using the SmartCat and Scopus databases, supplemented by snowballing from references in relevant studies. Keywords such as “research management platform,” “model management platform,” “healthcare research,” and “AI in healthcare” were used to identify pertinent studies. As new platforms were discovered, additional searches were conducted with terms such as “DSpace” and “ePrints” to gather further information about these specific systems. Additionally, snowballing was used to explore the references listed in the initially found literature to identify further relevant sources.

3 Requirements

This section outlines the requirements for the AI-Research Management Platform, LLACE, which were gathered through a comprehensive process involving interviews with researchers, analysis of existing literature, and examination of similar platforms. The primary goal was to identify the specific needs and challenges faced by researchers in the healthcare domain and design a platform that effectively addresses these requirements. Detailed notes from interviews with researchers are available in [Appendix A](#).

3.1 REQUIREMENTS GATHERING

The requirements gathering process involved several key activities. Initially, two rounds of in-depth interviews were conducted with healthcare professionals and researchers to gain a detailed understanding of their workflows, pain points, and desired features in a research management system. The first interview was an online meeting lasting approximately one hour with five researchers at UMCG, focusing on their specific needs for LLACE. The second interview was conducted with an individual at UMCG who has a strong understanding of the technologies currently in use, providing insights into potential integrations and technological constraints. Both interviews were semi-structured, allowing for comprehensive discussion based on prepared questions and additional relevant input from the participants (see [Appendix A](#)).

The literature review involved analyzing existing research and platforms, such as DSpace and ePrints, to identify their strengths and limitations. This review provided a foundational understanding of the current state-of-the-art in research management systems and informed the requirements for LLACE. The examination of similar platforms helped gather insights into best practices and common features, particularly focusing on the unique needs of the healthcare sector.

3.2 SYSTEM FUNCTIONALITIES

The LLACE platform is designed to serve as a comprehensive tool for managing AI research in the healthcare domain. Based on the gathered requirements, the system will offer the following core functionalities:

- **User Management:** The platform will support creating, updating, and deleting user accounts, along with managing user roles and permissions. This includes secure authentication using institutional Single Sign-on credentials.
- **Research Project Management:** Researchers will be able to create, update, and

manage research projects. This includes secure data storage, managing project collaborators, and linking projects to related models and publications.

- **AI Model Management:** The system will support the uploading, updating, and management of AI models, including storing detailed metadata, versioning, and linking to related projects and publications. Models can also be deployed to environments like UbiOps for execution (see "Overview of UbiOps" subsection).
- **Search Functionality:** LLACE will provide advanced search capabilities tailored to different types of users, including researchers, data scientists, project managers, and administrators. The system will offer dynamic filtering options to refine search results effectively, catering to the unique needs of each user group.
 - **Researchers:**
 - * Perform keyword searches and filter results by project status.
 - * Categorize and search by research topics, authors, and institutions.
 - * Utilize detailed metadata searches, including publication dates.
 - **Data Scientists:**
 - * Access datasets and AI models based on data types, formats, and sources.
 - * Search for models using performance metrics, version history, and usage scenarios.
 - **Project Managers:**
 - * Monitor project progress and status with filters for timelines and milestones.
 - * Track collaborator contributions and access project documentation.
 - **Administrators:**
 - * Manage users by searching for activities, access patterns, and system changes.
 - * Retrieve audit logs for compliance and thorough review processes.
- **Data Integration and Compliance:** The platform will integrate with institutional systems to synchronize user information and research data. It will also ensure compliance with data privacy regulations like GDPR, featuring robust security measures such as data encryption and audit logs.
- **User Interface:** A user-friendly interface will include separate pages for global search, project management, model management, and a dashboard for quick access to a user's research activities.

3.3 MoSCoW TECHNIQUE

To prioritize the project requirements, the MoSCoW prioritization technique was used, categorizing requirements into four groups: Must Have, Should Have, Could Have, and Won't Have [23]. This prioritization is for the ideal implementation of LLACE, recognizing that

not all requirements may be met in the Minimum Viable Product (MVP). Each requirement has been assigned an identification number for traceability, as shown in Table 1.

Must	Should	Could	Won't
------	--------	-------	-------

Table 1: Color coding schema for MoSCoW Requirements based on interviews and literature analysis (see Appendix A).

3.4 FUNCTIONAL REQUIREMENTS

The functional requirements define specific functionalities that the LLACE platform must provide to meet user needs. Each requirement is denoted by a unique identifier [F-#].

#	Requirement
F-1	User Management: The platform must allow for the creation, updating, and deletion of user accounts.
F-2	User Authentication: Secure user authentication using institutional Single Sign-on credentials is required to ensure the integrity and security of user data.
F-3	Role and Permission Management: The system must manage user roles and permissions to control access to different parts of the platform.
F-4	User Profile Management: The system must enable the creation, updating, and deletion of user profile information, including roles, affiliations, and contact details.
F-5	Research Project Management: The platform must allow users to create, update, and delete research projects, including the ability to upload and store research data securely.
F-6	Data Storage: Secure storage for research data must be provided, ensuring compliance with data privacy regulations such as GDPR.
F-7	Project-Project Linking: The platform must provide links to related research projects, enabling users to navigate easily between them.
F-8	Project-Model Linking: The platform must provide links to related AI models, facilitating easy navigation and access.
F-9	Project-Publication Linking: Research projects should link to related publications, facilitating access to relevant literature.
F-10	AI Model Management: The platform must allow for the uploading, updating, and deletion of AI models, including detailed metadata management.
F-11	Model Metadata Management: Store model metadata, including versioning, performance metrics, and links to relevant projects and publications.

Table 2: Functional Requirements based on interviews and literature analysis (see Appendix A).

#	Requirement
F-12	Model Deployment: The platform should enable the deployment of AI models to environments like UbiOps for execution (see "Overview of UbiOps" subsection).
F-13	Model-Project Linking: Link AI models to related projects, providing context and relevance to the models.
F-14	Model-Model Linking: AI models must provide links to other related models that are clickable, allowing users to see connections and relevance.
F-15	Model-Publication Linking: Link AI models to related publications, facilitating access to relevant literature.
F-16	Model Collaborator Management: Manage collaborators for AI models, allowing for role assignments and permissions.
F-17	Advanced Search: The platform must provide advanced search capabilities for research projects, AI models, publications, authors, datasets, and user profiles.
F-18	Search Filtering: Implement filtering options to refine search results based on specific criteria such as publication date, project status, and research area.
F-19	Institutional Repository Integration: The platform could integrate with institutional repositories to synchronize research data, providing a seamless experience for researchers.
F-20	Audit Logs: Could provide audit logs to track user activities and system changes, supporting transparency and accountability.
F-21	International Collaboration: Could support international collaboration features, including language localization and cross-border data sharing protocols.
F-22	Dedicated Pages: The platform could have dedicated pages for global search, browsing research projects, AI models, and managing user profiles.
F-23	Project Management Page: Must have a dedicated page for managing Research Projects, including tools for collaboration and data management.
F-24	Model Management Page: Must have a dedicated page for managing AI Models, including version control and metadata.
F-25	Account Management Page: Could have a dedicated page for managing user account information, including security settings and personal details.
F-26	User Dashboard: Should have a dashboard that provides an overview of a user's projects, models, and recent activities.
F-27	Login Page: Should have a secure login page with options for account recovery and support.

Table 3: Functional Requirements (continuation of Table 2), based on interviews and literature analysis (see Appendix A).

3.5 NON-FUNCTIONAL REQUIREMENTS

The non-functional requirements ensure the platform performs efficiently and is reliable, secure, and easy to use. Each requirement is denoted by a unique identifier [NF-#].

#	Requirement
NF-1	Performance: The system should handle concurrent access by multiple users without a response time exceeding 2 seconds for standard operations.
NF-2	Efficiency: Data retrieval and processing operations should be optimized to minimize delays, ensuring quick access to information.
NF-3	Scalability: The platform should scale to accommodate increasing amounts of data and users, supporting at minimum the number of staff and students at UMCG concurrently.
NF-4	Modularity: The system should be designed with modular components to facilitate maintenance and updates, allowing for easy replacement or upgrade of individual modules.
NF-5	Reliability: The system should maintain an uptime of 99.9%, with mechanisms for automatic failover and disaster recovery.
NF-6	Data Integrity: Could implement backup and recovery mechanisms to prevent data loss, ensuring that all data can be restored within 4 hours in the event of a failure.
NF-7	Usability: The user interface must be intuitive and easy to navigate, with user testing to ensure a positive user experience.
NF-8	Documentation: Provide comprehensive documentation and support for users.
NF-9	Data Security: Should ensure data encryption in transit and at rest, with regular security audits and vulnerability assessments.
NF-10	Maintainability: Must use best practices in coding and documentation to ensure the system is easy to maintain and extend.
NF-11	Accessibility: The pages for managing Research Projects, AI Models, and account information should be accessible with no more than 3 clicks from the homepage.

Table 4: Non-Functional Requirements based on interviews and literature analysis (see Appendix A).

4 Architecture

This section presents the architecture of LLACE, which comprises multiple layers and components, each serving a specific function within the platform. The architecture is illustrated through various diagrams, explaining the system’s interactions and components in varying levels of detail.

4.1 IDEAL ARCHITECTURE

The ideal architecture represents the comprehensive vision for LLACE, incorporating all planned features and components. This architecture aims to provide a robust, scalable, and secure platform for managing AI-driven research in healthcare.

4.1.1 SYSTEM CONTEXT DIAGRAM

The System Context Diagram (Figure 1) provides a high-level overview of the interactions between LLACE and its external entities, including users and external systems. This diagram illustrates how different components and users interact with LLACE to achieve a seamless AI-research management experience.

In this diagram, researchers and administrators are the primary users interacting with LLACE. Researchers upload, download, and manage AI models and data, and they request the execution of models on the UbiOps platform. Administrators manage user accounts, system settings, and monitor the overall system to ensure security, compliance, and efficient operation.

The AI-Research Management Platform, which forms the core of LLACE, communicates with several external systems. It integrates with Authentication Services provided by UMCG to ensure secure user authentication through Single Sign-On (SSO). This integration ensures that users have secure access to the platform based on institutional authentication mechanisms such as LDAP and OAuth.

LLACE also interacts with Institutional Repositories, primarily from UMCG, to fetch and store research data. This integration allows for seamless synchronization of research data, ensuring that researchers have access to the latest information and can upload their data securely.

Furthermore, the platform connects with External Data Sources, such as Pure, which provide essential data for AI model training and research, as well as relevant publication information. Pure is integrated into the ideal implementation of LLACE, as it is the tool UMCG uses to manage its publications. This integration helps incorporate necessary datasets and

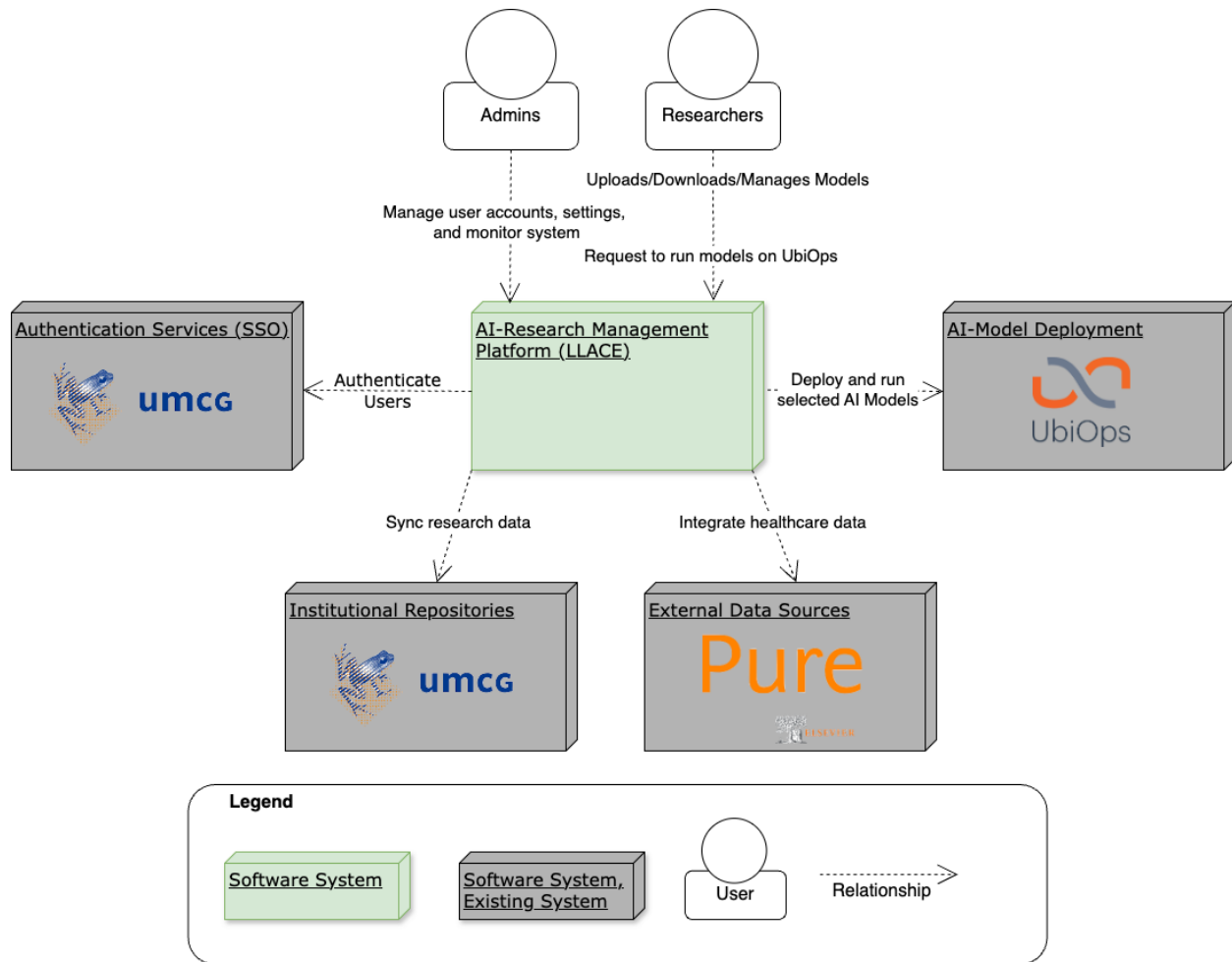


Figure 1: System Context Diagram

publications into the research process, enhancing the quality and relevance of the AI models being developed.

A crucial component of LLACE is its integration with the AI-Model Deployment service, specifically UbiOps (see [Overview of UbiOps](#)). UbiOps provides the computational resources and environment needed for deploying and running AI models. Researchers can request model executions on UbiOps directly through LLACE, ensuring that their models are deployed in a robust and scalable environment.

Overall, the interactions between the Web Application, Backend Services, and External Systems ensure that LLACE provides a comprehensive and efficient platform for managing AI research in the healthcare domain. By leveraging these integrations, LLACE supports researchers and administrators in their efforts to advance medical science through the effective management and deployment of AI models.

4.1.2 CONTAINER DIAGRAM

The Container Diagram (Figure 2) provides a high-level view of the major components or “containers” within LLACE, showcasing how they interact with each other and external systems. This diagram abstracts the internal workings of each container to focus on their roles and relationships.

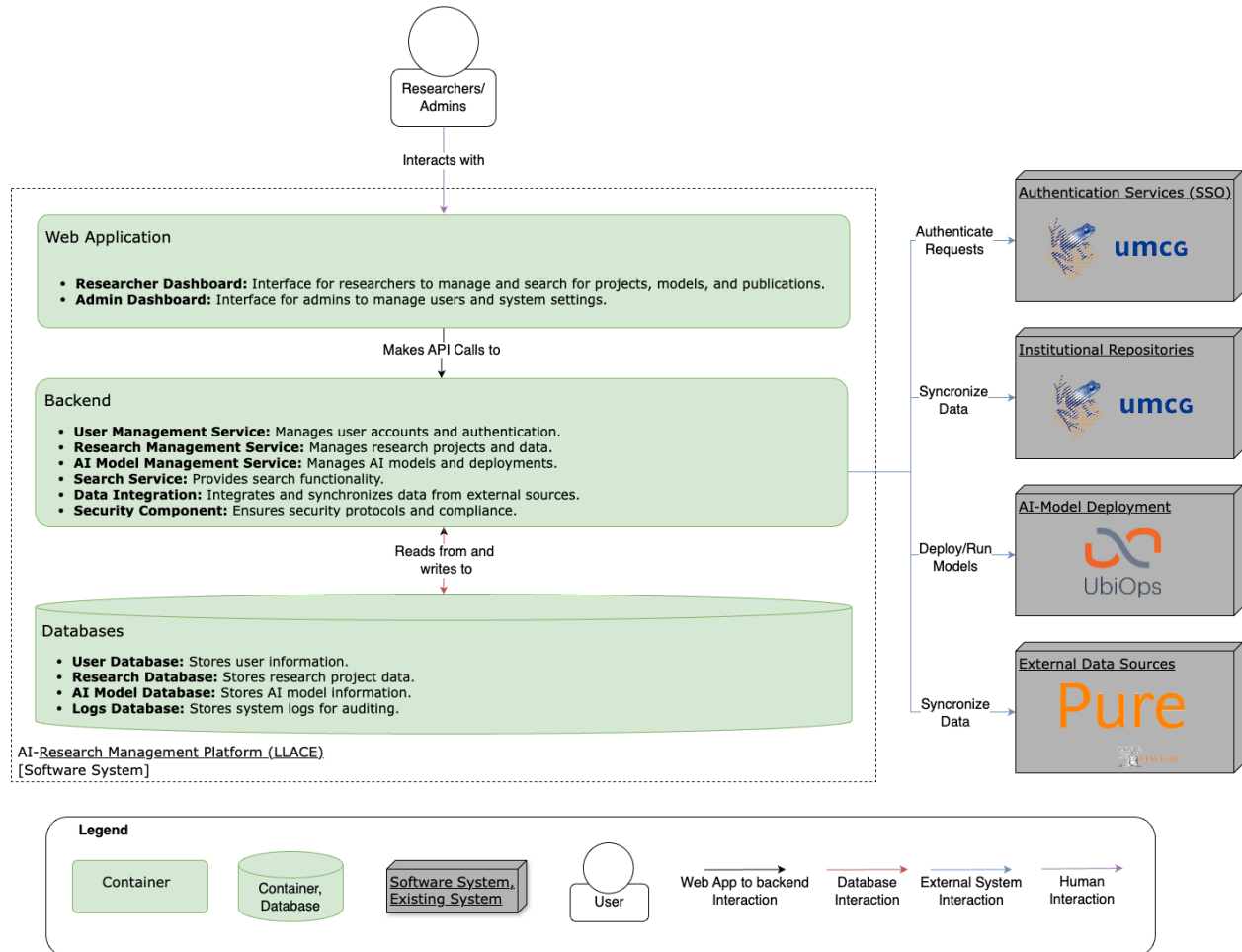


Figure 2: Container Diagram

The architecture is structured around several key containers:

- **Web Application:** Serves as the user interface for both researchers and administrators. Researchers use the application to manage and search for projects, models, and publications, while administrators use it to manage users and system settings. This application interacts with the Backend Services through API calls, handling various user requests and data operations.
- **Backend Services:** This container comprises multiple services responsible for different functionalities, such as User Management, Research Management, AI Model

Management, and Data Integration. These services communicate with each other and the Web Application using RESTful APIs, facilitating operations like CRUD (Create, Read, Update, Delete) on data and synchronization with external systems.

- **Database Layer:** Manages the persistent storage of data, including user information, research project data, AI model metadata, and system logs. The databases interact with the Backend Services to store and retrieve data as needed.
- **External Systems:** Includes UMCG’s Authentication Services for secure user authentication, Institutional Repositories for research data management, External Data Sources like Pure for additional research data, and UbiOps for deploying and running AI models.

The communication between these containers typically involves RESTful API calls, with HTTP as the primary protocol. This design allows for scalable and flexible interactions, enabling components to operate independently and communicate efficiently.

4.1.3 BACKEND COMPONENT DIAGRAM

The Backend Component Diagram (Figure 3) delves deeper into the internal structure of the Backend Services, detailing how different components within the backend interact and fulfill specific roles.

The backend is composed of several core components that work together to provide comprehensive functionality for LLACE. The **User Management Service** is responsible for handling all aspects of user authentication and authorization, ensuring secure access through integration with UMCG’s Authentication Services. This service manages user credentials, roles, and permissions, interfacing with the User Database to store and retrieve user-related information securely.

The **Research Management Service** plays a crucial role in managing research project data. It oversees the creation, updates, and deletion of projects, ensuring that data from Institutional Repositories and External Data Sources is accurately integrated and synchronized within LLACE. This service interacts with the Research Database to maintain comprehensive records of all research activities, enabling efficient data management and retrieval.

The **AI Model Management Service** is dedicated to managing AI models, including their metadata and versioning. This service facilitates the deployment of models on UbiOps, ensuring that models are run in a scalable and robust environment. By coordinating with the AI Model Database, it maintains detailed records of all AI models, supporting activities such as model training, testing, and deployment.

The **Data Integration Service** ensures that data from various external sources, such as Pure and other external data repositories, is effectively integrated into the platform. This

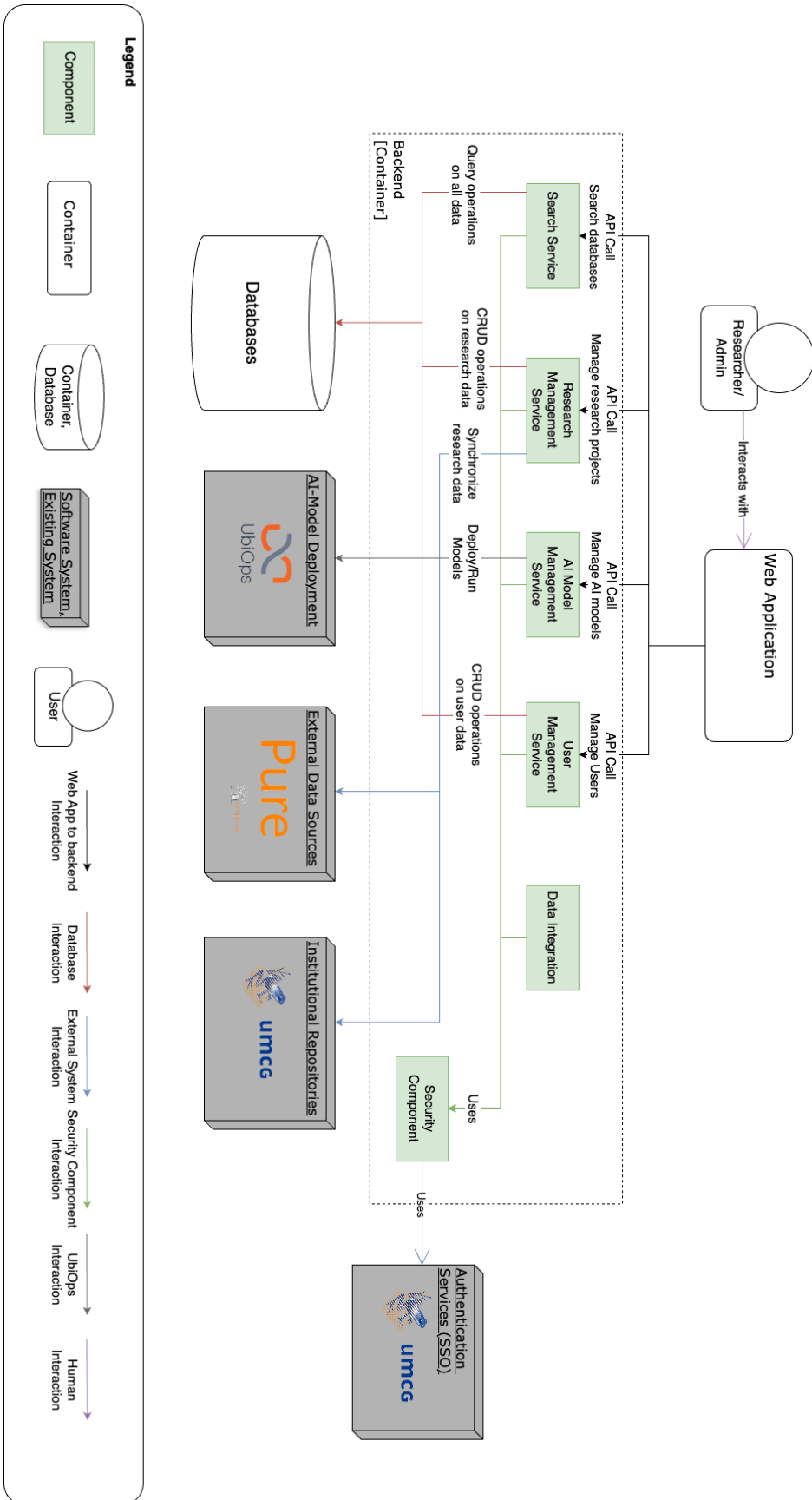


Figure 3: Backend Component Diagram

service plays a critical role in keeping the data up-to-date and consistent across the platform, thereby providing researchers with the most relevant and accurate information for their work.

Additionally, the **Search Service** enhances the usability of the platform by providing advanced search capabilities. It allows users to efficiently search across all databases, including user profiles, research projects, and AI models, facilitating easy access to the necessary information.

The **Security Component** is integral to the system, ensuring that all data interactions comply with established security protocols and regulatory requirements. It safeguards the platform's integrity by protecting against unauthorized access and data breaches, thereby maintaining the confidentiality and security of user data.

Lastly, the **Database Services** interface with the physical databases, handling the storage, retrieval, and management of all data. This includes user information, research project data, AI model metadata, and system logs, all of which are critical for the platform's operations.

These backend components communicate primarily via internal APIs, using secure communication protocols such as HTTPS to ensure data integrity and security. The modular design of these interactions allows for easy maintenance and scalability, supporting the platform's growth and adaptation to future needs.

4.1.4 IDEAL HIGH-LEVEL DIAGRAM

The Ideal High-Level Diagram (Figure 4) illustrates the comprehensive layered architecture of LLACE. This diagram highlights all the components and their interactions, showing how the ideal architecture connects various modules and external systems.

In this diagram, the architecture is divided into several layers:

- **Presentation Layer:** Utilizes React for the frontend interface, providing a user-friendly experience for researchers and administrators.
- **API Gateway Layer:** Employs NGINX to route requests to the appropriate services within the Business Layer, managing traffic and providing a single entry point to the system.
- **Business Layer:** Contains multiple modules, including Application Services, Integration, and Infrastructure. These modules manage user accounts, AI models, projects, and data integration.
- **Persistence Layer:** Uses Django ORM for data access and Redis for caching, ensuring efficient data storage and retrieval operations.
- **Database Layer:** Utilizes PostgreSQL for robust and scalable database management.

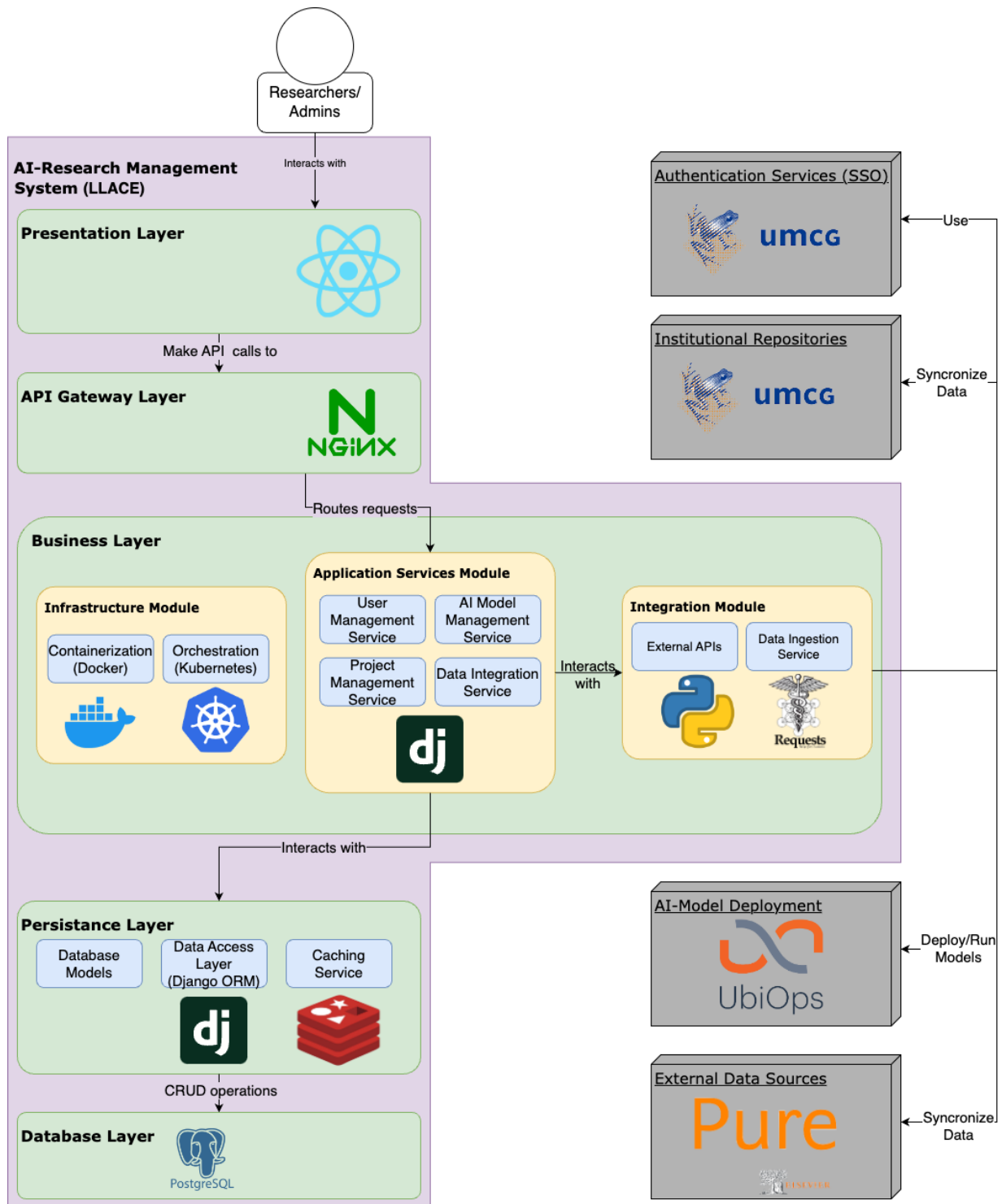


Figure 4: Ideal High-Level Diagram

This architecture is designed to provide a robust, scalable, and secure environment for managing AI-driven research in healthcare. It supports various integration points, ensuring that LLACE can interact with external systems, manage data efficiently, and provide a seamless user experience.

4.2 MVP ARCHITECTURE

Given the time and resource constraints, the MVP (Minimum Viable Product) architecture focuses on essential components and functionalities necessary to provide a working prototype of LLACE.

4.2.1 MVP HIGH-LEVEL DIAGRAM

The MVP High-Level Diagram (Figure 5) provides a simplified version of the ideal architecture, scoped down to ensure feasibility within the constraints of a Bachelor's thesis. This diagram focuses on the core components and interactions necessary to deliver a functional AI-research management system.

The MVP architecture is streamlined to prioritize essential features. The API Gateway and Infrastructure Module are omitted, and the integration with UMCG services is limited. The backend focuses on core functionalities, with PocketBase serving as the primary backend solution. PocketBase includes an embedded database, built-in authorization management, and a REST API, simplifying data management and user authentication.

The communication within the MVP architecture primarily involves RESTful API calls facilitated by Axios for data synchronization and management. This setup ensures that core functionalities, such as user management, AI model handling, and data integration, are efficiently managed within the limited scope of the MVP.

Interactions within the system are streamlined. Researchers and administrators interact with the system via the Presentation Module, which communicates with the Application Services Module to manage users, AI models, and research projects. The Application Services Module makes API calls to PocketBase to store, retrieve, and manage data. The Integration Module facilitates external data synchronization and AI model deployment through interactions with external APIs and UbiOps.

This MVP architecture ensures a functional and efficient platform for managing AI research in healthcare, while adhering to the constraints of the project. Further details on the technologies used will be provided in the Technology Stack section.

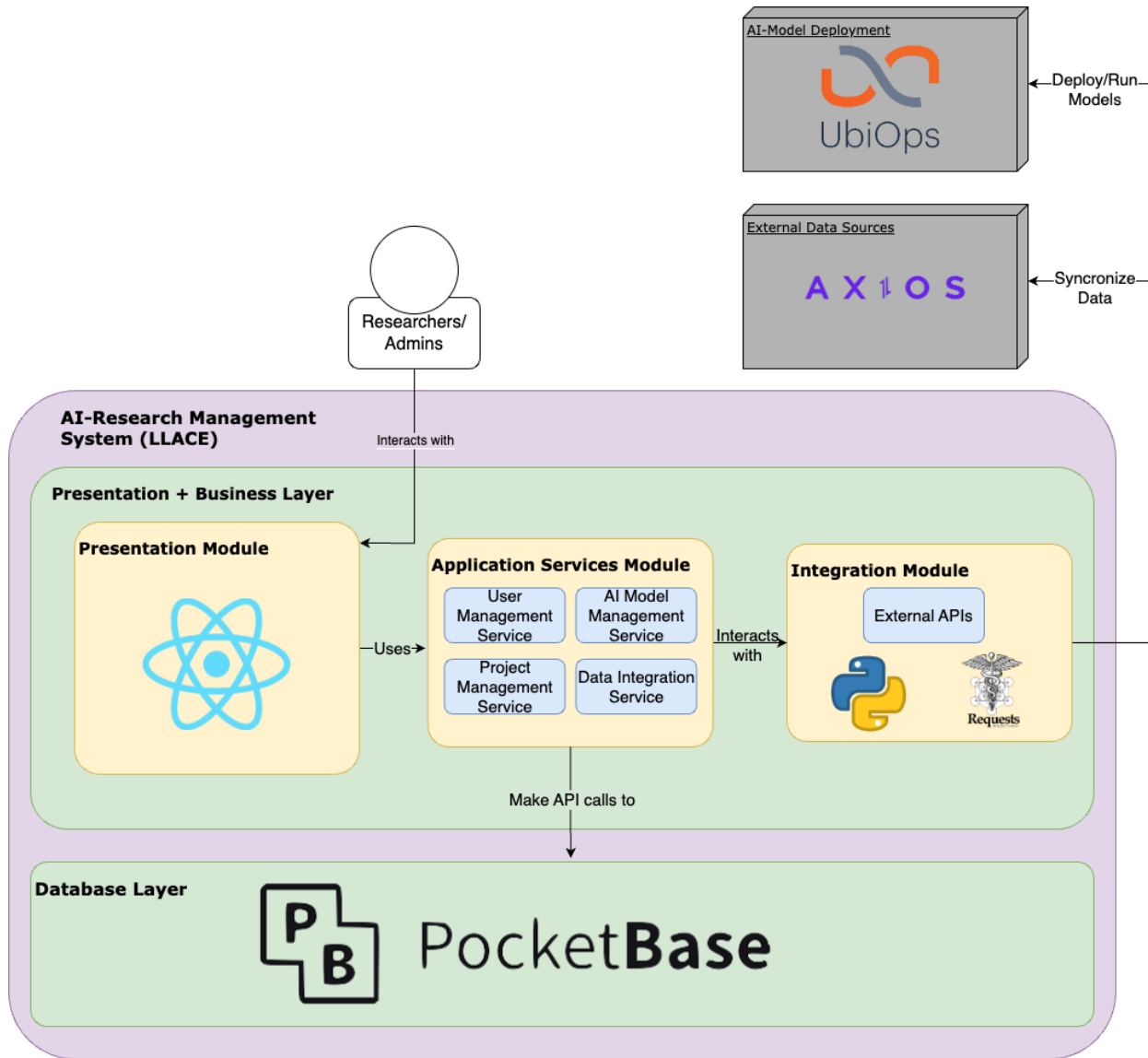


Figure 5: MVP High-Level Diagram

5 Design

The design of the LLACE platform emphasizes a modular and extensible architecture, ensuring that the system is robust, maintainable, and scalable. This section delves into the code design principles, focusing mainly on the frontend design due to the MVP scope and the use of PocketBase for handling most backend functionalities.

5.1 CODE DESIGN

The overall project is divided into frontend and backend modules. The frontend is developed using React, adhering to a standard React project structure. The code is organized into components (and subcomponents), pages, hooks, and context to promote clarity, simplicity, and maintainability. Each component is designed with a clear separation of concerns, allowing for isolated development, testing, and maintenance.

The backend consists of two main components: the PocketBase component, which handles most backend requests, and a Flask proxy-backend, which facilitates deployment requests to UbiOps. PocketBase is an open-source backend solution that includes an embedded database, built-in authorization management, a convenient dashboard UI, and a simple REST API. This setup ensures that data storage and management are efficiently handled, with the Flask backend serving specific purposes such as deployment requests.

The entire codebase for the LLACE platform, including both the frontend and backend modules, is maintained in an open-source repository on GitHub [5]. This accessibility not only facilitates future contributions from the community but also serves as a valuable resource for similar projects and other research initiatives .

5.2 FRONTEND DESIGN

The frontend design of LLACE is structured to provide a user-friendly interface for researchers and administrators. The design follows standard React project practices:

- **Components:** Reusable UI elements, each representing a specific part of the interface. These are further divided into subcomponents where necessary to promote reusability and maintainability.
- **Pages:** Individual views or screens of the application, composed of various components and subcomponents. Each page corresponds to a specific route in the application.
- **Hooks:** Custom hooks to encapsulate and reuse logic, such as data fetching and state management.
- **Context:** Provides a way to pass data through the component tree without having to pass props down manually at every level, facilitating state management across the application.

For detailed visuals of the frontend design, please refer to the appendix (Appendix B). The images in the appendix illustrate key pages and features, including the Account Page, Dashboard, Edit Project Page, General Search, Model Creation, Model Deployment, Project Creation, and Project Page.

5.3 MODULARITY

The design of LLACE emphasizes abstraction and modularity to simplify development and enhance maintainability. Each service within the Application Services Module is intended as a standalone module. This modular approach allows for working on individual components without affecting others, facilitating parallel development and reducing the risk of integration issues.

5.4 EXTENSIBILITY DECISIONS

Several key decisions were made to ensure the extensibility of the LLACE platform. All interactions between the frontend and backend, as well as between internal services, are conducted through RESTful APIs. This standardization facilitates the integration of new services and components, as well as external systems, without requiring significant changes to the existing codebase.

The use of PocketBase for backend services ensures that the backend of the system is robust and extensible. PocketBase provides an embedded database, built-in authorization management, and a simple REST API, allowing the platform to scale as new features and requirements emerge.

These design decisions collectively contribute to a robust, flexible, and maintainable platform, capable of evolving to meet future needs and requirements.

6 Methods

The development and implementation of the LLACE platform involved several key methodologies to ensure a robust and user-centered solution. This section outlines the methods used for data selection, technology stack selection, and the overall development process, including key design decisions and their rationale. The primary goal was to create a scalable, secure, and efficient platform tailored to the specific needs of healthcare researchers, particularly within the University Medical Center Groningen (UMCG).

6.1 DATA SELECTION

Data selection is crucial for developing a robust and effective AI-Research Management Platform. The LLACE platform will integrate various data sources to provide comprehensive data management and research capabilities. One of the primary data sources considered is Pure [7], a research information management system that aggregates data from various institutional repositories. This ensures that researchers have access to a wide range of high-quality data for their projects. Additionally, external data sources such as public datasets and institutional databases will be integrated to enhance the research capabilities and data diversity available within the platform.

6.2 TECHNOLOGY STACK

The development of the LLACE MVP involves utilizing a carefully selected technology stack that balances rapid development, ease of use, and scalability. The technology stack can be divided into two categories: the ideal technology stack for a fully developed platform and the technology stack used for the MVP.

6.2.1 IDEAL TECHNOLOGY STACK

The ideal technology stack for the fully developed LLACE platform includes:

- **Frontend:** React.js
- **API Gateway:** NGINX
- **Containerization:** Docker
- **Orchestration:** Kubernetes
- **Backend:** Django

- **Language: Python**
- **HTTP Requests: Python Requests Library**
- **Caching: Redis**
- **Database: PostgreSQL**

Technology	Rationale
React.js [37]	Fast development with reusable components, large community support, and excellent performance due to the virtual DOM.
NGINX [32]	High-performance HTTP server and reverse proxy, known for its stability, rich feature set, and simple configuration.
Docker [17]	Ensures consistency across development and production environments, facilitating reliable deployment.
Kubernetes [12]	Automates deployment, scaling, and management of containerized applications.
Django [10]	Provides a full-featured framework with ORM, authentication, and an admin interface, suitable for a data-centric application.
Python [35]	Offers simplicity and readability, with a large ecosystem of libraries and frameworks.
Python Requests Library [28]	Simplifies HTTP requests, making it easy to integrate with APIs.
Redis [30]	In-memory data structure store, used as a database, cache, and message broker, providing fast access to data.
PostgreSQL [15]	ACID compliance, strong support for complex queries, and robust performance.

Table 5: Ideal Technology Stack for LLACE

Alternatives Considered for Ideal Stack:

Frontend:

- **Vue.js [39]:** Offers a simpler learning curve and flexible integration but has a smaller community and less mature ecosystem compared to React.

- **Angular** [13]: A full-featured framework with strong support for enterprise applications, but its complexity makes it less suitable for rapid development.

Backend:

- **Express.js (Node.js)** [16]: Lightweight and allows for fast development using full-stack JavaScript. However, its less structured nature can lead to messy code without discipline.
- **Flask (Python)** [29]: Lightweight, flexible, and easy to learn but requires more setup for features that come out-of-the-box with Django.

Database:

- **MongoDB** [18]: Offers a schema-less design allowing flexibility and fast setup. However, it can lead to inconsistent data if not managed carefully and may not perform as well for complex queries compared to relational databases.
- **MySQL** [3]: Widely used and performs well for read-heavy operations but requires schema design upfront and is less flexible for rapid prototyping.

6.2.2 MVP TECHNOLOGY STACK

The MVP version of the LLACE platform focuses on essential components and functionalities. The chosen technologies are streamlined to ensure rapid development and ease of use.

Frontend: React.js

React.js [37] was selected for its fast development capabilities with reusable components, a large community, and excellent performance due to the virtual DOM. Its component-based architecture allows for modularity and reusability, essential for building a scalable and maintainable frontend.

Backend: PocketBase and Flask

The backend is simplified with PocketBase [26] handling most backend requests and a Flask proxy-backend for making deployment requests to UbiOps [34]. PocketBase is an open-source backend solution with an embedded database, built-in authorization management, a convenient dashboard UI, and a simple REST API, ensuring efficient data storage and management. Flask [29] is used to provide a lightweight, easy-to-set-up backend for specific deployment functionalities.

HTTP Requests: Axios

Due to the unavailability of PURE API keys, Axios [41] was used to handle HTTP requests for fetching publications and other external data sources. Axios is a promise-based HTTP client for the browser and Node.js, providing an easy-to-use API for making asynchronous HTTP requests.

Technology	Rationale
React.js [37]	Fast development with reusable components, large community support, and excellent performance due to the virtual DOM.
PocketBase [26]	Provides an embedded database, built-in authorization management, and a simple REST API, acting as the primary backend.
Flask [29]	Lightweight and easy to set up, used for specific backend functionalities like deployment requests to UbiOps.
Axios [41]	Promise-based HTTP client for making asynchronous HTTP requests, used for fetching external data due to the unavailability of PURE API keys.

Table 6: MVP Technology Stack for LLACE

6.3 OVERVIEW OF UBIOPS

6.3.1 INTRODUCTION TO UBIOPS

UbiOps [34] is a robust deployment and operationalization platform for machine learning models and data science projects. It offers a streamlined process for deploying, managing, and scaling models in a production environment. The platform is designed to handle complex workflows and integrations, making it a preferred choice for institutions like the University Medical Center Groningen (UMCG).

6.3.2 WHY UBIOPS

UbiOps was specifically chosen by UMCG based on feedback from researchers actively using the platform. The decision to use UbiOps was influenced by its ability to integrate well with existing tools and workflows, which is crucial for large institutions with established processes. UbiOps provides a user-friendly interface and comprehensive documentation, facilitating easy adoption and use by researchers and data scientists.

Additionally, UbiOps is designed to support large-scale deployments, offering the necessary infrastructure to scale models efficiently. This scalability is particularly important for healthcare applications, where the volume of data and the complexity of models can be significant. Researchers at UMCG have highlighted these features as key benefits, making UbiOps an optimal choice for the platform [34].

Moreover, UbiOps offers robust security features and ensures high reliability, critical aspects in the healthcare domain where patient data privacy and regulatory compliance are paramount. The platform's capabilities align with UMCG's needs for secure, reliable, and scalable infrastructure for managing and deploying machine learning models.

6.3.3 FUNCTIONALITIES OF UBIOPS

UbiOps [34] provides a wide range of functionalities that support the deployment and management of machine learning models. One of the key functionalities is model deployment, which simplifies the process of deploying models into production, handling all underlying infrastructure.

It also supports the creation and management of complex data pipelines, enabling the automation of data processing and model workflows. Additionally, UbiOps offers detailed monitoring and logging features, which are essential for tracking model performance and debugging.

Furthermore, UbiOps facilitates version control for models and data pipelines, allowing easy rollback to previous versions if needed. These functionalities collectively make UbiOps a comprehensive platform for managing the entire lifecycle of machine learning models.

6.3.4 ALTERNATIVES

While there are several other tools available for model deployment and management, UbiOps was chosen for its comprehensive feature set and alignment with UMCG's requirements. Although AWS SageMaker [1] offers extensive features for model deployment and management, its complexity and the need for specialized AWS knowledge were barriers. UbiOps provides a more straightforward interface that better suits the needs of UMCG's research teams [34].

Google AI Platform [14] is another powerful tool, but it can be overly complex and expensive for certain use cases. UbiOps provides a cost-effective and simpler solution. Similarly, Azure ML [22] has robust capabilities but requires deep integration with the Azure ecosystem. UbiOps provides more flexibility with integrations, making it a better fit for UMCG's diverse infrastructure.

Lastly, while Kubeflow [27] is open-source and highly customizable, it requires significant setup and maintenance effort. UbiOps offers managed services that reduce the operational

burden on UMCG's IT team. These considerations highlight why UbiOps is the preferred choice for UMCG over other available tools [34].

7 Evaluation

This evaluation assesses the extent to which the LLACE platform meets the outlined functional and non-functional requirements, recognizing that it is an MVP (Minimum Viable Product) and not the final ideal implementation. The evaluation is based on initial testing, which was conducted to verify that the requirements were met correctly, but does not include comprehensive testing. It is important to note that due to time constraints, the platform was not tested by researchers at UMCG. Involving them in future testing phases could provide valuable insights, particularly regarding usability and practical functionality, and would be beneficial for refining the design and implementation.

7.1 FUNCTIONAL REQUIREMENTS

The following tables summarize the status of each functional requirement, indicating whether it is met or not, and providing brief notes on how the requirement is met or why it is not. The assessment was conducted through manual testing, where we verified the presence and functionality of the features against the specified requirements.

Requirement ID	Met	Notes
F-1	Yes	User accounts can be created, updated, and deleted.
F-2	No	Only a simple login page is implemented; institutional Single Sign-on not integrated.
F-3	Partially	Permissions are checked; more work needed to improve roles.
F-4	Yes	Users can manage profile information, excluding roles.
F-5	Yes	Comprehensive management of research projects, including data storage and updates.
F-6	Partially	Secure storage for research data provided; more work needed to secure data.
F-7	Yes	Links between related research projects are established.
F-8	Yes	Links between related AI models are provided.
F-9	Yes	Projects link to related publications; however, this should ideally be done using Pure.
F-10	Yes	Full support for uploading, updating, and managing AI models.
F-11	Yes	Model metadata is managed, including versioning and links to projects/publications.

Table 7: Evaluation of Functional Requirements

Requirement ID	Met	Notes
F-12	Yes	Model deployment to environments like UbiOps is supported.
F-13	Yes	Links from AI models to related projects are provided.
F-14	Yes	AI models link to other related models.
F-15	Yes	Models link to related publications; however, this should ideally be done using Pure.
F-16	Yes	Model collaborator management is implemented.
F-17	Yes	Advanced search capabilities are available for various platform components.
F-18	Yes	Search filtering options are implemented; filtering could be more intricate.
F-19	No	Institutional repository integration not implemented; planned for the ideal implementation of LLACE.
F-20	No	Audit logs not present; planned for the ideal implementation of LLACE.
F-21	No	International collaboration features not implemented, but possible to add easily in the future.
F-22	Yes	Dedicated pages for global search, project browsing, and profile management are provided.
F-23	Yes	A dedicated page for managing research projects is available.
F-24	Yes	A dedicated page for managing AI models is available.
F-25	Yes	A dedicated page for account management exists; requires UMCG integration for full functionality.
F-26	Yes	A dashboard provides an overview of user projects and models.
F-27	Yes	A secure login page is implemented.

Table 8: Evaluation of Functional Requirements (continuation of Table 7)

7.2 NON-FUNCTIONAL REQUIREMENTS

The MVP does not fully address the non-functional requirements due to its scope, but some aspects were considered. The evaluation of these requirements was based on qualitative assessments, considering the design choices and implementation strategies employed during development.

Requirement ID	Met	Notes
NF-1	Partially	Performance and response times are good, but not formally tested.
NF-2	Partially	Data retrieval is efficient; however, further optimization is needed.
NF-3	Partially	Designed to support UMCG staff and students; scalability needs further testing.
NF-4	Yes	Modular architecture facilitates maintenance and upgrades.
NF-5	No	Reliability goals like 99.9% uptime not yet validated; requires robust infrastructure.
NF-6	No	Backup and recovery mechanisms are not yet implemented.
NF-7	Yes	The user interface is user-friendly, with good design practices.
NF-8	Yes	Documentation is provided through this thesis, GitHub, and a demo video.
NF-9	Partially	Basic security measures like username and password implemented; most security provided by PocketBase.
NF-10	Yes	Best practices in coding and documentation are followed.
NF-11	Yes	Key pages are easily accessible within the user interface.

Table 9: Evaluation of Non-Functional Requirements

The non-functional requirements, including performance, scalability, and security, were only partially addressed in this MVP version due to the constraints of the project. Further testing, especially involving real users from UMCG, would be crucial in validating these aspects and refining the platform to meet high standards. These users can provide practical feedback, particularly on usability and the overall user experience, which is essential for ensuring that the platform meets the needs of its intended audience.

In conclusion, the LLACE MVP meets many of the critical functional requirements, providing a solid foundation for managing AI research projects and models. Some advanced features and non-functional requirements are not fully addressed, highlighting areas for further development and refinement in future iterations. The platform is well-positioned for continued enhancement, with a focus on extending functionality and improving performance, security, and scalability.

In addition to the detailed architectural and functional descriptions provided in this thesis, a comprehensive demonstration of the LLACE platform’s capabilities is available in the recorded demo [6]. This video provides a visual walkthrough of key features, including user account management, project and model handling, and search functionalities.

8 Conclusion

The integration of Artificial Intelligence (AI) into healthcare presents a significant opportunity for advancements in diagnosis, treatment, and operational efficiency. However, managing AI-driven research poses unique challenges, such as ensuring data privacy, achieving high accuracy, and integrating diverse data sources. This thesis addresses these challenges by proposing the design and implementation of an AI Research Management Platform, LLACE, tailored specifically for the healthcare domain.

The motivation for LLACE arises from the critical need for a robust system that not only manages AI model deployment but also aligns with the ethical and operational standards required in healthcare research. The platform is designed to support the lifecycle of AI models, from development and validation to deployment and monitoring, thereby enhancing collaboration and streamlining research processes at the University Medical Center Groningen (UMCG).

The design and implementation of LLACE were guided by a comprehensive understanding of user requirements, gathered through interviews with UMCG researchers and an analysis of existing literature. The MVP of LLACE employs a technology stack that includes React.js for the frontend and PocketBase for backend management. These technologies were chosen for their balance of rapid development, ease of use, and scalability. The integration with UbiOps for model deployment and the use of Axios for handling HTTP requests are critical components that ensure the platform's flexibility and adaptability.

The architecture of LLACE, both ideal and MVP, showcases a modular and extensible design, ensuring that the platform can evolve to meet future needs. The emphasis on modularity and abstraction in the code design supports maintainability and scalability, which are essential for the platform's long-term success. The modular structure facilitates isolated development, testing, and maintenance, reducing integration risks and supporting parallel development.

Future work will focus on enhancing the platform's capabilities, including improving model deployment features, integrating performance metrics, automating tag generation, and refining search functionalities. Additional efforts will be directed toward incorporating more detailed information on research projects and models, gathering continuous user feedback, and enhancing security measures. These enhancements aim to solidify LLACE as a comprehensive and indispensable tool for healthcare research.

In conclusion, LLACE represents a significant step forward in managing AI-driven research in healthcare. The platform addresses the specific needs of healthcare researchers by providing a centralized system for efficient project management, fostering collaboration, and driving advancements in medical science. As shown, LLACE overcomes the limitations of other similar platforms. The detailed architecture and design decisions outlined in this thesis set a strong foundation for LLACE's future development and broader adoption, positioning it as a pivotal tool in the intersection of AI and healthcare.

9 Future Work

The current implementation of the LLACE platform as an MVP has achieved its primary goal and laid a solid foundation for managing AI-driven research projects. However, several enhancements and features can be incorporated to improve its functionality, usability, and overall effectiveness. This section outlines the potential future work for the platform, detailing how these features can be integrated at the design and architecture level.

9.1 MODEL DEPLOYMENT AND PERFORMANCE METRICS

One significant improvement is the enhancement of model deployment capabilities. Currently, models can be deployed to UbiOps, but future iterations should focus on gathering and displaying results from deployed models. Integrating performance metrics, such as accuracy, precision, recall, and F1 scores, directly into the platform will provide researchers with immediate insights into model effectiveness. These metrics can be integrated into the existing *AI Model Management Service* and displayed on the model details page in the frontend. This would involve extending the backend to collect and store these metrics, potentially requiring additional data pipelines and modifications to the Database Layer to store performance data.

9.2 AUTOMATED TAG GENERATION

To streamline the organization of models and research projects, an automated tag generation system can be implemented. By analyzing related publications and other relevant data sources, the platform can automatically generate and assign tags to models and projects. This automation can be achieved using Natural Language Processing (NLP) techniques integrated into the *Data Integration Service*. The service would extract keywords and themes from publications, aiding in the generation of accurate and relevant tags. These tags would then be stored in the appropriate database tables and displayed on the frontend, enhancing the search and filtering capabilities.

9.3 ADVANCED SEARCH CAPABILITIES

The current search functionality in LLACE is limited to basic keyword searches. Future improvements should include advanced search capabilities, such as filtering options based on various criteria like publication year, research area, author, institution, and project duration. Implementing these features would involve enhancing the *Search Service* to support more complex queries and introducing a semantic search layer to understand and interpret user queries better. The frontend would need to be updated to provide users with customizable search filters, improving the user experience.

9.4 ENHANCED INFORMATION FOR PROJECTS AND MODELS

Future versions of LLACE should consider adding more detailed information to research project and model pages. This could include project duration, funding details, milestones, and collaboration opportunities. Enhancing these pages would require extending the data schema in the *Research Database* and *AI Model Database* to store additional fields. Moreover, the *Research Management Service* and *AI Model Management Service* would need updates to handle these new data points. Conducting further interviews with researchers will help identify the most beneficial information to include, ensuring the platform evolves to meet user needs.

9.5 USER FEEDBACK AND ITERATIVE IMPROVEMENTS

Establishing a robust feedback mechanism is crucial for continuous improvement. Implementing a feature within the platform to allow users to submit feedback will help identify pain points and areas for enhancement. This feedback system can be integrated into the *Web Application* layer, with data collected and stored in a dedicated feedback database. The *User Management Service* can manage access and permissions related to this feedback, ensuring data is secure and used effectively for iterative improvements.

9.6 INTEGRATION WITH ADDITIONAL DATA SOURCES

Expanding the integration capabilities of LLACE to include more external data sources will significantly enhance the platform's utility. Integrating additional institutional repositories, public datasets, and other research management systems will provide a comprehensive hub for accessing diverse data. This integration will require extending the *Data Integration Service* to handle new data formats and sources. The architecture should support adding connectors for these sources, ensuring data is integrated seamlessly and consistently.

9.7 SCALABILITY AND PERFORMANCE OPTIMIZATION

As the platform grows, ensuring scalability and optimizing performance will be critical. Future work should focus on improving the underlying infrastructure to handle increased user load and data volume. This involves optimizing database queries, enhancing caching mechanisms, and considering the use of distributed systems to manage large-scale operations efficiently. The *Database Layer* might need to be scaled out, and load balancing strategies should be implemented within the *API Gateway Layer* to manage traffic effectively.

9.8 SECURITY AND COMPLIANCE ENHANCEMENTS

With increasing focus on data privacy and security, future iterations of LLACE should prioritize enhancing security measures. This includes implementing advanced authentication mechanisms such as multi-factor authentication and ensuring compliance with data protection regulations like GDPR. The *Security Component* would need to be expanded to include these features, incorporating regular security audits and updates to maintain platform integrity and trustworthiness.

In conclusion, the future work outlined above will not only enhance the functionality and usability of LLACE but also ensure it meets the high standards expected in modern research environments. These improvements will require coordinated updates across multiple components of the platform, ensuring that the system remains robust, scalable, and secure as it evolves.

10 Acknowledgments

I would like to express my deepest gratitude to my supervisors, Prof. Dimka Karastoyanova and Michel Medema, MSc., for their invaluable guidance and support throughout this project. Their expertise and insights have been instrumental in shaping the direction and success of this thesis.

I am also profoundly grateful to the researchers and staff at the University Medical Center Groningen (UMCG) who provided their time, knowledge, and feedback. Their contributions have been crucial in understanding the practical requirements and challenges faced by healthcare researchers, ensuring that the LLACE platform meets their needs effectively.

Finally, I would like to thank everyone who has supported me throughout my degree and the thesis process. Your encouragement and belief in my work have been a tremendous source of strength and motivation.

REFERENCES

- [1] Inc. Amazon Web Services. *AWS SageMaker: Machine Learning Model Building*. 2017. Accessed: 2024-06-07.
- [2] Ricardo Carvalho Amorim, João Aguiar Castro, João Rocha da Silva, and Cristina Ribeiro. A comparison of research data management platforms: architecture, flexible metadata and interoperability. *Universal Access in the Information Society*, 16:851–862, 2017.
- [3] Oracle Corporation. *MySQL: The world’s most popular open source database*. 1995. Accessed: 2024-06-07.
- [4] Somalee Datta, Jose Posada, Garrick Olson, Wencheng Li, Ciaran O’Reilly, Deepa Balraj, Joseph Mesterhazy, Joseph Pallas, Priyamvada Desai, and Nigam Shah. A new paradigm for accelerating clinical data science at stanford medicine, Mar 2020.
- [5] Oscar de Francesca. Llace: Ai research management platform. <https://github.com/oscardef/AI-Research-Management-Platform>, 2024. Accessed: 2024-07-27.
- [6] Oscar de Francesca. Llace platform demonstration. <https://youtu.be/IAQFAHJF15w>, 2024. Accessed: 2024-07-27.
- [7] Elsevier. Pure - research information management system. <https://www.elsevier.com/products/pure>. Accessed: 2024-06-07.
- [8] ePrints. eprints - open access repository solutions. <https://www.eprints.org/uk/>. Accessed: 2024-07-27.
- [9] Facebook Engineering. Introducing fblearner flow: Facebook’s ai backbone. <https://code.facebook.com/posts/1072626246134461/introducing-fblearner-flow-facebook-s-ai-backbone/>, May 2016. Accessed: 2024-04-02.
- [10] Django Software Foundation. *Django: The web framework for perfectionists with deadlines*. 2005. Accessed: 2024-06-07.
- [11] Kristina K. Gagalova, M. Angelica Leon Elizalde, Elodie Portales-Casamar, and Matthias Görge. What you need to know before implementing a clinical research data warehouse: Comparative review of integrated data repositories in health care institutions. *JMIR Formative Research*, 4(8), 2020. Open Access; Green Open Access.
- [12] Google. *Kubernetes: Automating deployment, scaling, and management of containerized applications*. 2014. Accessed: 2024-06-07.
- [13] Google. *Angular: One framework. Mobile desktop*. 2016. Accessed: 2024-06-07.

-
- [14] Google. *Google AI Platform: AI and Machine Learning Products*. 2019. Accessed: 2024-06-07.
- [15] PostgreSQL Global Development Group. *PostgreSQL: The world's most advanced open source relational database*. 1996. Accessed: 2024-06-07.
- [16] TJ Holowaychuk and StrongLoop. *Express - Node.js web application framework*. 2010. Accessed: 2024-06-07.
- [17] Docker Inc. *Docker: Empowering App Development for Developers*. 2013. Accessed: 2024-06-07.
- [18] MongoDB Inc. *MongoDB: The database for modern applications*. 2009. Accessed: 2024-06-07.
- [19] Christopher J. Kelly, Alan Karthikesalingam, Mustafa Suleyman, Greg Corrado, and Dominic King. Key challenges for delivering clinical impact with artificial intelligence. *BMC Medicine*, 17(195), October 2019.
- [20] Jihyun Kim. Finding documents in a digital institutional repository: Dspace and eprints. In *Proceedings of the Association for Information Science and Technology*, 2005.
- [21] LYRASIS. Dspace. <https://dspace.lyrasis.org/>. Accessed: 2024-07-27.
- [22] Microsoft. *Azure Machine Learning: Enterprise-grade machine learning service*. 2018. Accessed: 2024-06-07.
- [23] E. Miranda. Moscow rules: A quantitative exposé. In *Lecture Notes in Business Information Processing*, pages 19–34. Springer International Publishing, 2022.
- [24] University of Groningen. Habrok hpc cluster. <https://www.rug.nl/society-business/centre-for-information-technology/research/services/hpc/facilities/habrok-hpc-cluster?lang=en>. Accessed: 2024-07-26.
- [25] Philip R. O. Payne, Tara B. Borlawsky, William Stephens, Matthew C. Barrett, Tri Nguyen-Pham, and Andrew W. Greaves. The triton project: Design and implementation of an integrative translational research information management platform. *AMIA ... Annual Symposium proceedings / AMIA Symposium*, 2010:617–621, 2010.
- [26] PocketBase. Pocketbase: Open-source backend solution. <https://pocketbase.io/>. Accessed: 2024-07-30.
- [27] Kubeflow Project. *Kubeflow: The machine learning toolkit for Kubernetes*. 2018. Accessed: 2024-06-07.
- [28] Kenneth Reitz and Requests Development Team. *Requests: HTTP for Humans*. 2011. Accessed: 2024-06-07.

-
- [29] Armin Ronacher and Pallets Projects. *Flask: A micro web framework for Python*. 2010. Accessed: 2024-06-07.
- [30] Salvatore Sanfilippo. *Redis: An open source, in-memory data structure store, used as a database, cache, and message broker*. 2009. Accessed: 2024-06-07.
- [31] SAS. Sas model manager. https://www.sas.com/en_us/software/model-manager.html, 2024. Accessed: 2024-04-02.
- [32] Igor Sysoev. Nginx – high-performance http server and reverse proxy. 2004. Accessed: 2024-06-07.
- [33] Uber Engineering. Meet michelangelo: Uber’s machine learning platform. <https://eng.uber.com/michelangelo/>, September 2017. Accessed: 2024-04-02.
- [34] UbiOps. Ubiops - deployment and operationalization platform for machine learning models. <https://ubiops.com/>. Accessed: 2024-06-07.
- [35] Guido van Rossum and Python Software Foundation. *Python: A programming language that lets you work quickly and integrate systems more effectively*. 1991. Accessed: 2024-06-07.
- [36] M. Vartak and S. Madden. Modeldb: Opportunities and challenges in managing machine learning models. *IEEE Data Eng. Bulletin*, 41(4):16–25, 2018.
- [37] Jordan Walke and Facebook. React – a javascript library for building user interfaces. 2013. Accessed: 2024-06-07.
- [38] Christian Weber and P. Reimann. Mmp - a platform to manage machine learning models in industry 4.0 environments. In *IEEE 24th International Enterprise Distributed Object Computing Workshop (EDOCW)*. IEEE, October 2020. Available: <https://doi.org/10.1109/EDOCW49879.2020.00025>.
- [39] Evan You. *Vue.js: The Progressive JavaScript Framework*. 2014. Accessed: 2024-06-07.
- [40] Kun-Hsing Yu, Andrew L. Beam, and Isaac S. Kohane. Artificial intelligence in health-care. *Nature Biomedical Engineering*, 2(10):719–731, October 2018.
- [41] Matt Zabriskie. *Axios: Promise based HTTP client for the browser and node.js*. 2014. Accessed: 2024-06-07.
- [42] Matei Zaharia, Andrew Chen, Aaron Davidson, Ali Ghodsi, Sue Ann Hong, Andy Konwinski, Siddharth Murching, Tomas Nykodym, Paul Ogilvie, Mani Parkhe, et al. Accelerating the machine learning lifecycle with mlflow. *IEEE Data Eng. Bull.*, 41(4):39–45, 2018.
- [43] Ömer Dalkıran, İdil Aker, Semanur Öztemiz, Zehra Taşkın, and Sevgi Koyuncu Tunç. Usability testing of digital libraries: The experience of eprints. *Procedia - Social and Behavioral Sciences*, 147:535–543, 2014.

A MEETING NOTES: RESEARCHER REQUIREMENTS GATHERING - ANONYMIZED

- **Introduction:**

- One of the researchers was not able to attend the meeting.
- Welcoming remarks and introduction by myself, explaining the thesis focus on developing an AI-Research Management Platform for the healthcare domain.

- **Understanding of the Project:**

- The primary goal is to develop a robust, user-friendly platform supporting the entire lifecycle of AI models in healthcare research, from data integration and model evaluation to deployment and collaboration.

- **Anticipated System Requirements:**

- User Management: Secure accounts and access control.
- Persistence Storage: A database for AI research data and models.
- Profile Storage: Managing researcher profiles, potentially integrated with user management for security purposes.
- Search Functionality: Advanced tools for locating research, models, and colleagues.
- Research Registration: Easy process for uploading and registering projects.
- AI Algorithm/Model Repository: Manage diverse AI algorithms and models.

- **Feedback and Scope Clarification:**

- The system should support (or allow for future) international collaboration and keep this capability in mind as a precondition.
- Suggestion to create a visual aid to help stakeholders understand the platform.

- **Detailed Workflow Descriptions by Researchers:**

- PhD students' workflow from data collection to cleaning, emphasizing the need for effective data handling and integration with existing healthcare data environments.
- Current use of diverse software for data operations and the local storage of models and computations in tools like VSCode.

- **Challenges and Improvements Suggested by a Team Member:**

- Initial project definitions often lack specificity, making early stages of AI projects feel aimless.

-
- The platform could offer structured pipelines to guide projects, aiding in defining clearer goals and methodologies.

- **Standardization and Evaluation Needs:**

- A standardized approach to conducting and sharing research would be beneficial.
- Emphasis on detailed data descriptions and evaluation metrics to ensure models' applicability to different data sets and conditions.
- Potential for a standardized model form detailing use cases, data types, and evaluation results.

- **Management and Collaboration:**

- Discussion on who would manage model metadata and generalization to ensure models are broadly applicable.
- Importance of linking projects, especially for building upon previous research or validating existing models.

- **Security and Privacy Concerns:**

- Ensuring data safety and privacy within the healthcare institution's digital boundaries.
- Compliance with ISO 27001 and GDPR.
- Consideration of institution-specific data security rules and the need for project-specific ethical approvals.


- **Platform Functionality and Integration Needs:**

- Desire for Git-like functionality for research registration and updates.
- Importance of filtering capabilities in search functions for tasks, departments, diseases, and data types.
- Integration with current tools and data formats predominantly used at the institution, such as Python, R, and Excel.
- Use of platforms like Ubiops and Azure noted, with the potential for integration or avoiding functional overlap.


- **Future Considerations and Suggestions:**

- The potential for an "AI-Hub" to facilitate collaboration on AI-related projects.
- Importance of not duplicating efforts with existing tools like Ubiops or conflicting with established infrastructure.

B LLACE FRONTEND DESIGN ILLUSTRATIONS

SEARCH PROFILES RESEARCH PROJECTS MODELS 

My Account

CHANGE PROFILE PICTURE

Name
Oscar de Francesca

Username
oscar

Institution
University of Groningen

Department
Computer Science

Research Interests

- AI
- Healthcare

Publications

- [Computer science](#)

[EDIT PROFILE](#)

Figure 6: Account Page: Users can view and edit their profile information.

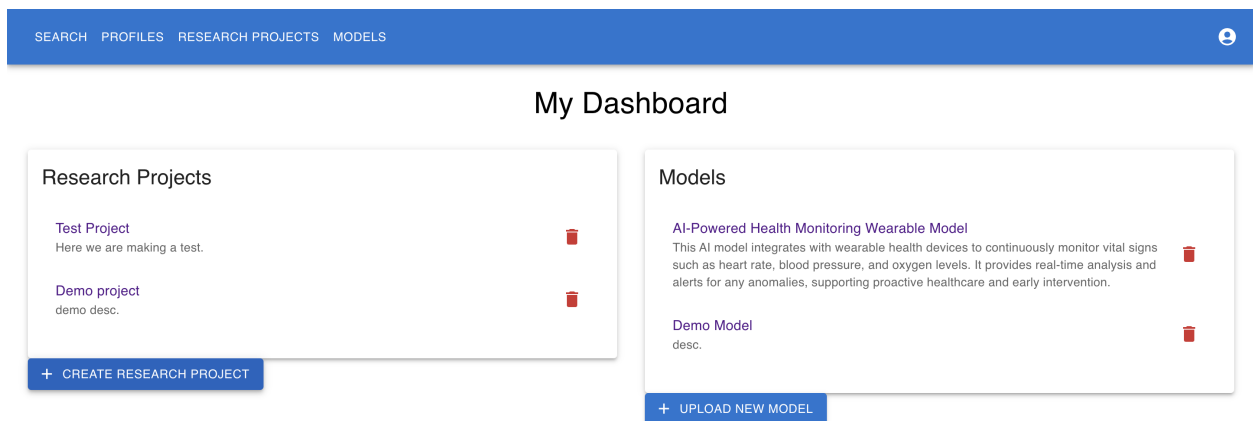


Figure 7: Dashboard: Users can access various functionalities such as managing and creating Research Projects and AI models.

SEARCH PROFILES RESEARCH PROJECTS MODELS

AI-Driven Climate Models

Description

Developing advanced AI models to predict climate change scenarios. This project aims to leverage AI techniques to analyze and predict future climate patterns based on historical and real-time data. The models developed will help in understanding the impacts of climate change and aid in the creation of mitigation strategies.

Status

Active

Tags

AI Climate Change Environment Data Analysis

New Tag

Details

Detail Key	Detail Value	
goal	Improve accuracy of clir	
duration	3 years	
budget	\$1,000,000	

Collaborators

- Diana White
- Bob Brown
- Charlie Green
- George Lewis
- Fiona Clark

Related Projects

- AI for Genomic Data Analysis
Applying AI techniques to analyze genomic data for...
- AI for Drug Discovery
Utilizing AI to accelerate the drug discovery proc...
- AI in Surgical Robotics
Exploring the use of AI in enhancing surgical robo...

Related Models

- AI-Powered Chronic Disease Management Model
Focused on managing chronic diseases such as hyper...
- AI-Driven Diabetes Management Model
Designed to assist in managing diabetes, this AI m...
- AI-Enhanced Drug Interaction Model
This model uses AI to predict potential drug inter...

Figure 8: Edit Project Page: Users can modify details of their research projects.

SEARCH PROFILES RESEARCH PROJECTS MODELS

Search
Healthcare

FILTER BY

- Profiles
- Models
- Research Projects
- Publications

Models

[AI-Driven Mental Health Assessment Model](#) AI Mental Health
This AI model assesses mental health conditions by analyzing behavioral data and self-reported symptoms. It aims to support mental health professionals by providing insights into the severity of conditions such as depression and anxiety, and recommending personalized treatment plans.
Collaborators: Alice Johnson, Hannah Walker, Diana White, George Lewis

[AI-Powered Chronic Disease Management Model](#) AI Chronic Disease Management Healthcare
Focused on managing chronic diseases such as hypertension and asthma, this AI model provides personalized recommendations by analyzing patient data from wearable devices and health records. The model aims to improve patient compliance and health outcomes through continuous monitoring and adaptive feedback.
Collaborators: George Lewis, Fiona Clark, John Doe
Status: active
Version: v2.1

[AI-Driven Diabetes Management Model](#) AI Diabetes Management Healthcare
Designed to assist in managing diabetes, this AI model analyzes continuous glucose monitoring data and other health metrics to provide personalized recommendations. It helps patients and healthcare providers make informed decisions regarding insulin dosage, diet, and lifestyle changes.
Collaborators: Alice Johnson, Jane Smith, George Lewis, Ethan Harris, Fiona Clark
Status: active
Version: v1.2

[AI-Assisted Radiology Interpretation Model](#) AI Radiology Diagnostics Healthcare
Developed to support radiologists, this AI model interprets medical imaging data, including X-rays, CT scans, and MRIs. It provides preliminary analysis and highlights areas of concern, thereby speeding up the diagnostic process and reducing the workload on radiologists.
Collaborators: Charlie Green, Alice Johnson, Hannah Walker, John Doe
Status: active
Version: v1.0

Figure 9: General Search: Provides powerful search capabilities across projects, models, profiles, and publications.

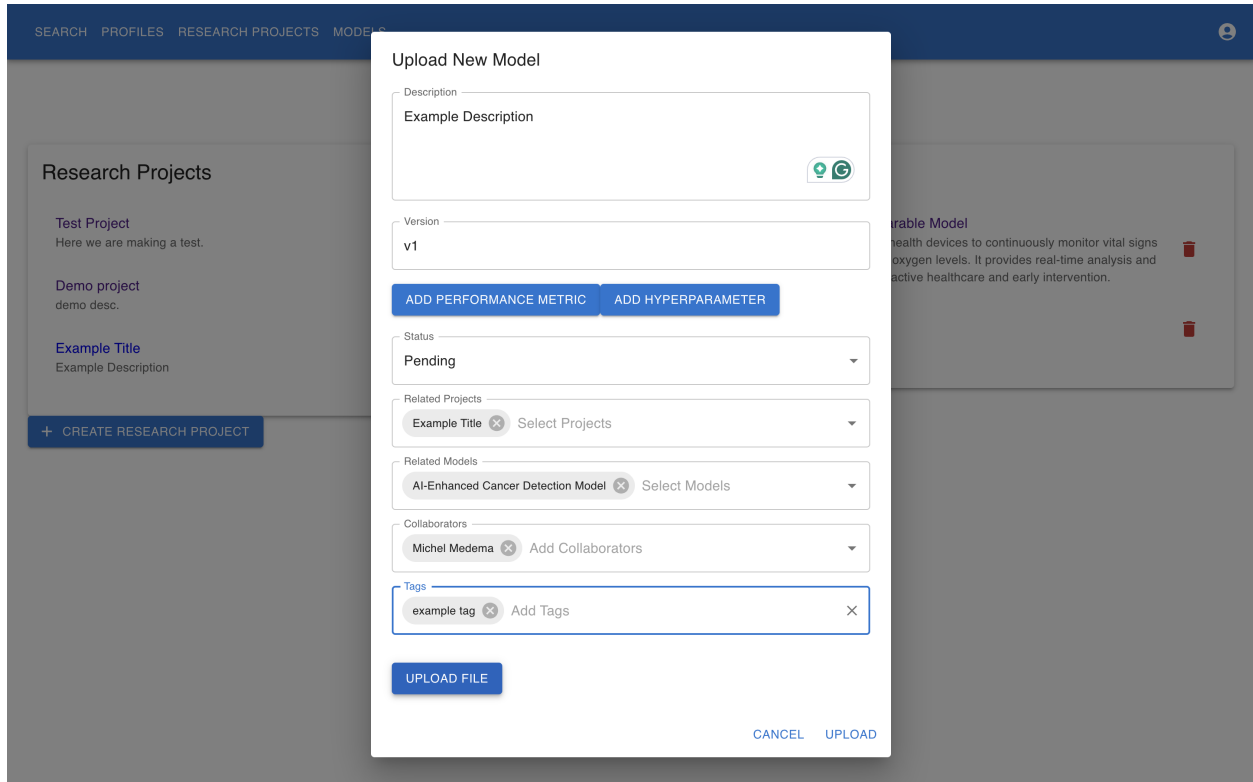


Figure 10: Model Creation: Users can upload and configure new AI models.

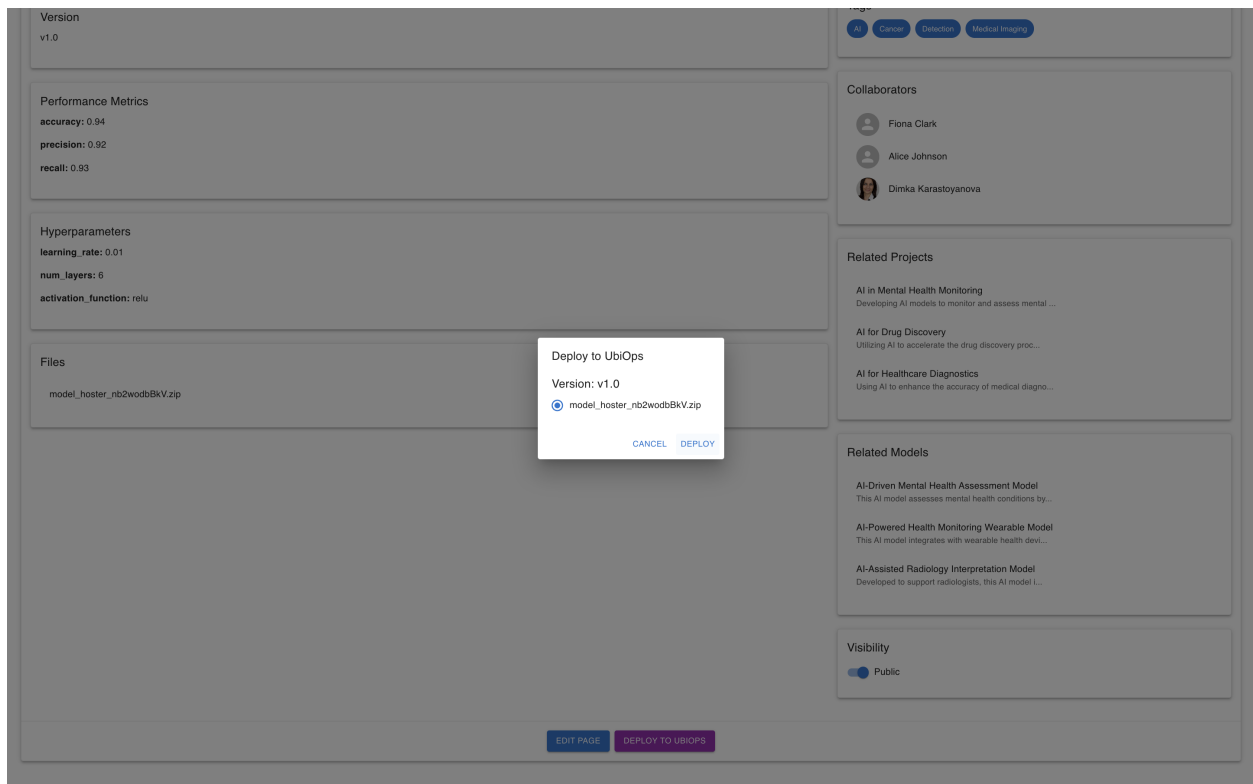


Figure 11: Model Deployment: Facilitates deploying models to UbiOps.

The image shows a 'Create Research Project' modal form overlaid on a dashboard. The dashboard background includes a navigation bar with 'SEARCH', 'PROFILES', 'RESEARCH PROJECTS', and 'MODELS', and a sidebar with 'Research Projects' containing 'Test Project' and 'Demo project'. The modal form has the following sections:

- Project Title:** A text input field containing 'Example Title'.
- Description:** A text area containing 'Example Description' with a small icon on the right.
- Status:** A dropdown menu currently set to 'Complete'.
- Tags:** A field containing 'example tag' and 'example tag 2' with 'Add Tags' text.
- Action Buttons:** 'ADD DETAIL' and 'ADD DATA SOURCE' buttons.
- Related Projects:** A dropdown menu showing 'AI-Driven Climate Models' and 'AI for Remote Patient Monitoring'.
- Related Models:** A dropdown menu showing 'AI-Powered Health Monitoring Wearable Model' and 'AI-Driven Diabetes Management Model' with 'Select Models' text.
- Collaborators:** A dropdown menu showing 'Dimka Karastoyanova' and 'Add Collaborators' text.

At the bottom right of the modal are 'CANCEL' and 'CREATE' buttons.

Figure 12: Project Creation: Users can create new research projects.

The screenshot shows a project page for "AI-Driven Climate Models". At the top, there is a blue navigation bar with the text "SEARCH PROFILES RESEARCH PROJECTS MODELS" and a user profile icon. The main content area is divided into several sections:

- Description:** Developing advanced AI models to predict climate change scenarios. This project aims to leverage AI techniques to analyze and predict future climate patterns based on historical and real-time data. The models developed will help in understanding the impacts of climate change and aid in the creation of mitigation strategies.
- Status:** active (indicated by a green tag).
- Details:**
 - goal: Improve accuracy of climate predictions.
 - duration: 3 years
 - budget: \$1,000,000
- Collaborators:** A list of five people with profile icons: Diana White, Bob Brown, Charlie Green, George Lewis, and Fiona Clark.
- Tags:** AI, Climate Change, Environment, Data Analysis.
- Related Projects:**
 - AI for Genomic Data Analysis: Applying AI techniques to analyze genomic data for...
 - AI for Drug Discovery: Utilizing AI to accelerate the drug discovery proc...
 - AI in Surgical Robotics: Exploring the use of AI in enhancing surgical robo...
- Related Models:**
 - AI-Powered Chronic Disease Management Model: Focused on managing chronic diseases such as hyper...
 - AI-Driven Diabetes Management Model: Designed to assist in managing diabetes, this AI m...
 - AI-Enhanced Drug Interaction Model: This model uses AI to predict potential drug inter...
- Related Publications:**
 - AI in Climate Prediction: Source: Climate Journal
 - Impact of AI on Climate Models: Source: Environmental Science

Figure 13: Project Page: Provides detailed information about a specific project, including title, description, additional details, collaborators, and related models, projects, and publications.