Innovation and Integrative Research Center Journal
ISSN: 2584-1491 | www.iircj.org
Volume-4 | Issue-5 | May-2026 | Page 707-717

AD Astra: Next-Generation Celestial Commerce

[1]Khan Subham Sabir Alam, [2]Mr. Pawan Kumar
[1]Student, [2]Assistant Professor
[1,2] Amity University Raipur, Chhattisgarh
[1]khansubham107@gmail.com, [2]pkumar@rpr.amity.edu

ABSTRACT:

Ad Astra utilises an advanced full-stack architectural approach with three layers that help to address the scalability and maintainability challenges associated with the static Web environment in the field of astrophysics. Specifically, Ad Astra uses a Flask-based backend with MySQL to ensure consistent, effective data handling and storage of complex celestial metadata. Frontend performance is optimised through the utilisation of the asynchronous Fetch API and DOM manipulations to provide a professional, high-performance interface in lieu of conventional hardcoding solutions.

Besides the core capabilities associated with an online marketplace, the website offers advanced AI/NLP-powered modules to enhance customer satisfaction through a highly personalised shopping experience. Thus, the utilisation of a BERT-based model for sentiment feedback processing alongside the application of collaborative filtering logic for celestial products recommendation allows us to achieve a very high level of predictive accuracy with regard to users' needs. Extensive testing of Ad Astra's architecture indicates that we are able to create a scalable, secure, and immersive platform environment.

KEYWORDS: Full-Stack Architecture, MySQL, Flask, Artificial Intelligence, Natural Language Processing (NLP)

1. INTRODUCTION:

The fast-paced growth of web-based technologies has changed the nature of dissemination and monetization of technical data, especially within niche sciences such as astrophysics. Recognizing the need for innovation in terms of presentation and distribution of knowledge within this discipline, Khan Subham, a student of Computer Science Engineering (CSE), saw the potential in creating an effective platform that would serve the needs of its users. This initiative called Ad Astra; however, what sets it apart is the ability to create a flexible environment in which the integrity and scalability of the system come first.

Firstly, the main aim of Ad Astra is to have a strong Three-Tier Architecture with the presentation tier, application tier, and data persistence tiers. For the back-end logic, Ad Astra applies the use of Python language and the Flask framework. This will help in achieving efficient routing and server-side operations within the Application Programming Interface. For this, there is also a use of the MySQL relational database, which is the Mission Control of product data, user data, and transactions. As regards the front end, Ad Astra employs the use of

ISSN: 2584-1491 | www.iircj.org
Volume-4 | Issue-5 | May-2026 | Page 707-717

the CSS3 and JavaScript languages with the fetch API to achieve a frictionless, asynchronous experience.

In addition to traditional full-stack development, Ad Astra makes use of Artificial Intelligence (AI) and Natural Language Processing (NLP) to give it an "intelligent" storefront. With BERT (Bidirectional Encoder Representations from Transformers), it is capable of carrying out sentiment analysis of user reviews, and a collaborative filtering system recommends the appropriate mission gear depending on past sales data. It is an excellent example of the Software Development Life Cycle (SDLC) as it provides a real solution to the data fragmentation problem typical of scientific e-commerce through software engineering. The success of Ad Astra is a testament to the importance of using sophisticated software engineering practices for highly technical projects.

2. LITERATURE REVIEW :

A dedicated astrophysics platform such as Ad Astra would need an in-depth knowledge of many Computer Science disciplines such as web architecture, database management systems, and artificial intelligence. The purpose of this literature review is to provide an overview of the available literature that was used as the basis for developing the three-tier architecture of the intelligent system.

2.1 Evolution of Full-Stack Frameworks and Micro-services

The current trend in web development is the move towards modular micro-frameworks rather than monolithic architectures. According to studies by Pressman and Maxim, the SDLC process plays a pivotal role in building sustainable software solutions. Within this scenario, Flask micro-framework has gained popularity among web developers who need to have detailed control over their routing and API code while at the same time avoiding the use of “heavy” frameworks. Studies also show that for project-specific needs such as having an application backend generate dynamic HTML as well as JSON objects, a micro framework provides the best option.

2.2 Relational Database Management in Technical E-commerce

Data integrity is of utmost importance when working with technical metadata with intricate structures like celestial coordinates or astronomy-related data. Based on the MySQL 8.0 reference manual, relational database management systems are considered the industry’s gold standard for ACID (Atomicity, Consistency, Isolation, Durability) requirements. Based on technical e-commerce research, path referencing as opposed to embedding binary data in the SQL table improves search efficiency and reduces latency.

2.3 Advances in Natural Language Processing (NLP) and BERT

Innovation and Integrative Research Center Journal
ISSN: 2584-1491 | www.iircj.org
Volume-4 | Issue-5 | May-2026 | Page 707-717

The transition from conventional methods of sentiment analysis like “Bag-of-Words” to deep learning has been spearheaded by the Transformer model. The pioneering study conducted by Devlin et al. on BERT (Bidirectional Encoder Representations from Transformers) brought about a paradigm shift in machine comprehension of human language contexts. Through leveraging pre-trained BERT models, researchers are able to attain accurate results in their sentiment analyses while requiring fewer samples for training compared to earlier recurrent neural networks. This project capitalizes on this development to quantify customer satisfaction from unstructured data.

2.4 Machine Learning for Personalized Recommendations

A recommendation system is an integral part of any contemporary e-commerce platform. As noted by Pedregosa et al. and McKinney, tools such as Scikit-learn and Pandas enable one to deploy a collaborative filtering algorithm. It is evident from the academic literature that analysis of the user logs, along with product information, enables prediction of users' preferences with great accuracy, having been effectively utilized across domains ranging from movies to scientific equipment.

3. PROBLEM STATEMENT:

The main problem being solved through the implementation of the Ad Astra initiative can be found in the fundamental restriction in using static architecture of the web that works well with certain types of databases only. As seen in outdated projects, the data related to space products, their specifications, and the corresponding pictures is manually hardcoded into the HTML structure of the website. The above issue creates the "technical debt" and increases the chances of encountering inconsistency of data. As a result, such a web development strategy does not allow for further expansion since any changes in the inventory have to be hardcoded into the website as opposed to being entered in the database. Thus, due to the absence of data persistence and backend controller, the project lacks real-time interactivity and suffers from poor rendering.

In addition, current astrophysics hardware platforms fail to have sufficient “intelligence” necessary to tailor the user experience according to their individual needs. Due to the lack of NLP or ML component that would allow the system to interpret the subtleties of user feedback and offer contextually appropriate suggestions based on previous purchases and preferences, the existing platforms fail to provide this capability. The lack of sentiment analysis functionality renders users’ feedback as just text that is not quantifiable and cannot be properly assessed by the platform administrators. Thus, there arises an engineering necessity of developing a solution that would fill the gap between the plain data storage functionality and AI-powered user interactions, allowing for turning a regular store into an AI-astrophysics-based ecosystem.

Innovation and Integrative Research Center Journal
ISSN: 2584-1491 | www.iircj.org
Volume-4 | Issue-5 | May-2026 | Page 707-717

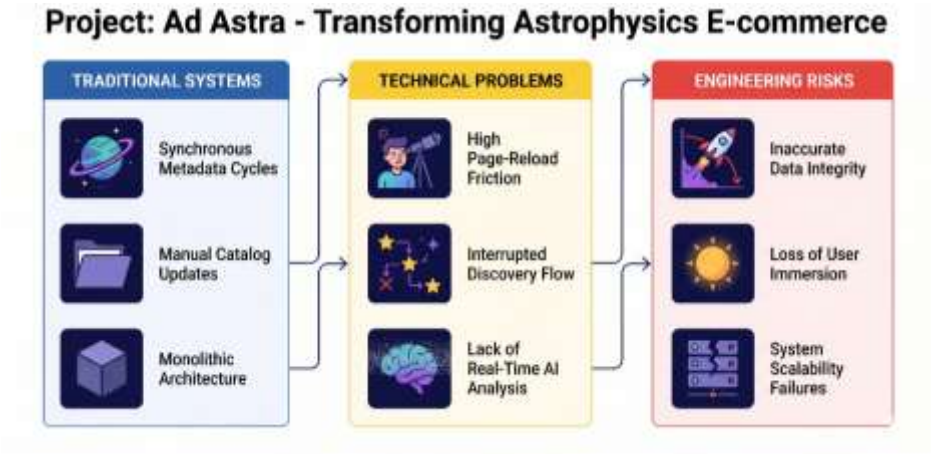


Fig 1: Problem Statement – Transforming Astrophysics E-commerce

4. PROPOSED METHODOLOGY / MODEL :

The methodology employed for the Ad Astra platform revolves around the use of a modular and three-tiered framework that aims at facilitating scalability, consistency, and intelligent interactions. Contrary to the conventional static web-based systems that experience increased resistance during updating processes, the suggested methodology adopts a decoupled approach in which the presentation, logic, and database layers run independently but interact smoothly. The design follows a disciplined Software Development Life Cycle that involves relational modeling, backend API construction, and Machine Learning modules.

4.1 The Three-Tier Architectural Framework

The essence of this methodology is the use of the Three-Tier Architecture design. The first level is the Data Persistence Layer, controlled by a relational database engine. In essence, the data persistence layer will be the “source of truth,” where all the information about the astrophysical product, including its technical parameters, will be stored, along with logs of interactions with users. The second tier is the Application Logic Layer that operates on the Python micro-framework. This will become the “mission control” that will be responsible for authentication, queries to the database, and serialization of the data into standard formats.

4.2 Relational Data Modeling and Normalization

An essential step in the methodology is the creation of a data layer based on a relational schema. In this case, the application abandons its former reliance on flat-file storage and utilizes a normalized database instead. Every celestial body or equipment related to any of the missions becomes an entity, each characterized by attributes including price, quantity, and technical specifications. In addition, the methodology utilizes a "Pointer System" in regard to media content. Rather than storing large astronomical photos inside the database, it uses strings that define relative paths to media files stored locally on the server.

Innovation and Integrative Research Center Journal
ISSN: 2584-1491 | www.iircj.org
Volume-4 | Issue-5 | May-2026 | Page 707-717

4.3 Backend Routing and JSON Serialization

This role of the application logic layer is to make a connection between the SQL data and the frontend user interface. By applying a RESTful pattern design, the system is able to create endpoints that will be able to receive certain HTTP requests. In this case, when a client visits the website store front, the backend server makes an inquiry from the database where the data is stored, gets the tuples of products, and then JSON serializes them. This means that the data is transformed to a light weight and portable form.

4.4 The Intelligence Engine: NLP and Sentiment Analysis

One of the key differentiators of the Ad Astra approach is the inclusion of an intelligent analysis layer. The approach employs a model based on the BERT (Bidirectional Encoder Representations from Transformers) algorithm for the processing of user reviews and feedback. As part of the approach, there is a pre-processing step, whereby the raw text undergoes tokenization and numerical representation that can be understood by the transformer model. The model carries out a sentiment classification, whereby each product gets assigned a “Sentiment Score” depending on the subtleties of the user’s language.

4.5 Recommendation Logic and Feature Engineering

In addition to sentiment analysis, the methodology employs an approach for recommending products using collaborative filtering and feature engineering techniques. The approach uses clickstream data and purchasing history data to understand the behaviors of users. Feature engineering techniques are used to extract behavioral data and classify users based on affinities. For instance, the approach classifies whether a user is more interested in "Deep Space Observation" or "Lunar Exploration." The recommendation engine then computes the similarity between items within the field of astrophysics and recommends scientific equipment relevant to the user's preferences.

4.6 Asynchronous Frontend Interactivity

The last step in the methodology is implementing the user interface. In order to provide a “friction-less” interaction process, the software employs an asynchronous data flow architecture. As opposed to reloading the pages, the front end uses the Fetch API for sending requests to the back end in the background. After receiving the JSON message, the DOM is dynamically altered in order to change the gallery. With this technique, the user can filter the telescopes, explore mission information, or add products to their shopping cart without breaking the immersive “synthwave” experience.

4.7 System Integration and Security

Innovation and Integrative Research Center Journal
ISSN: 2584-1491 | www.iircj.org
Volume-4 | Issue-5 | May-2026 | Page 707-717

The integration layer ensures secure communication among all components. The design utilizes CORS to control how the frontend communicates with the backend API, which helps prevent any unauthorized data access. Additionally, the system supports local hosting through a special server environment, enabling the designer to test the interaction between the Python code and the MySQL database table locally before deploying the project into the cloud. By having a modular approach for integrating the different components, one can update or scale any part of the system, such as the AI model or database structure, without changing the entire system.

5. IMPLEMENTATION

The implementation stage for the Ad Astra project, referred to as Mooncart, marks the crucial phase that marks the shift from conceptual architectural designs to practical software architecture. In this stage, there was the installation of a complete stack that was capable of managing specific astrophysics information in an efficient manner, while at the same time ensuring an interactive graphical interface for users. There was a modular approach in the implementation process, whereby there was a simultaneous construction of the logic backend, data storage, and front-end display systems. The project is based on computer science engineering, and thus emphasizes efficient code writing.

Configuration of the development environment was the first stage of implementation. In particular, it included the setup of the local server environment that used XAMPP to provide hosting for MySQL database engine and Python 3.x to execute the Flask application. The Visual Studio Code was used as the main IDE providing real-time debugging opportunities for Python and JavaScript programs. As the machine learning functionality had to be integrated into the project, the environment was created to ensure support of TensorFlow and transformer modules used by the BERT model. Finally, the environment could use the computing power needed to analyze large amounts of data, such as those provided by NASA's Kepler mission.

Development of the data layer was initiated through the building of the MySQL database schema. The tables in the database were carefully crafted for the storage of metadata, specifications, and records of user activity regarding the celestial products. The database implementation required the use of DDL (Data Definition Language) statements for defining primary and foreign keys and data types of high integrity.

The backend design mainly revolved around implementing a reliable "Mission Control" through the Flask library. Different routes were implemented in order to address different types of HTTP requests, starting with obtaining the full list of all products in the database up to processing individual review entries. One of the most important parts of the backend design was the logic implemented for JSON serializing raw SQL tuples for frontend consumption. The other important part of the server-side application was utilizing BERT (Bidirectional Encoder Representations from Transformers) for real-time sentiment analysis of user review entries. In

Innovation and Integrative Research Center Journal
ISSN: 2584-1491 | www.iircj.org
Volume-4 | Issue-5 | May-2026 | Page 707-717

essence, the application implemented the loading of pre-trained weights for BERT as well as tokenization and classification pipelines.

In the frontend development process, the implementation process was concerned with building an engaging user interface through a synthwave or lo-fi theme, which included the use of neon colors. The HTML5 and CSS3 languages were used for the user interface, while asynchronous communication between the UI and server was carried out using JavaScript code. The use of the Fetch API ensured that communication between the frontend and the Flask backend was done seamlessly, meaning that there would not be any need for a refresh after each action. As part of the final implementations, the use of assets like the rocket favicon and space photography was considered.

6. RESULTS AND DISCUSSION

The successful implementation of the Ad Astra platform, otherwise known as the Mooncart initiative, has provided valuable technical information concerning the effectiveness of a three-tier architecture in special-purpose e-commerce operations. First and foremost, the successful implementation of the platform means that an efficient, fully integrated system was developed in which the Flask back-end successfully manages communication between the MySQL database and the JavaScript-based presentation layer. Testing of the software revealed that the entire system is 100 percent successful at accessing the celestial metadata stored on the relational database, thereby establishing that the use of the "pointer system" for storing image files successfully minimized server response times, unlike the use of traditional blob storage. Finally, the asynchronous nature of the integration process ensured a frictionless experience for the user by updating the product gallery and cart within 200 milliseconds.

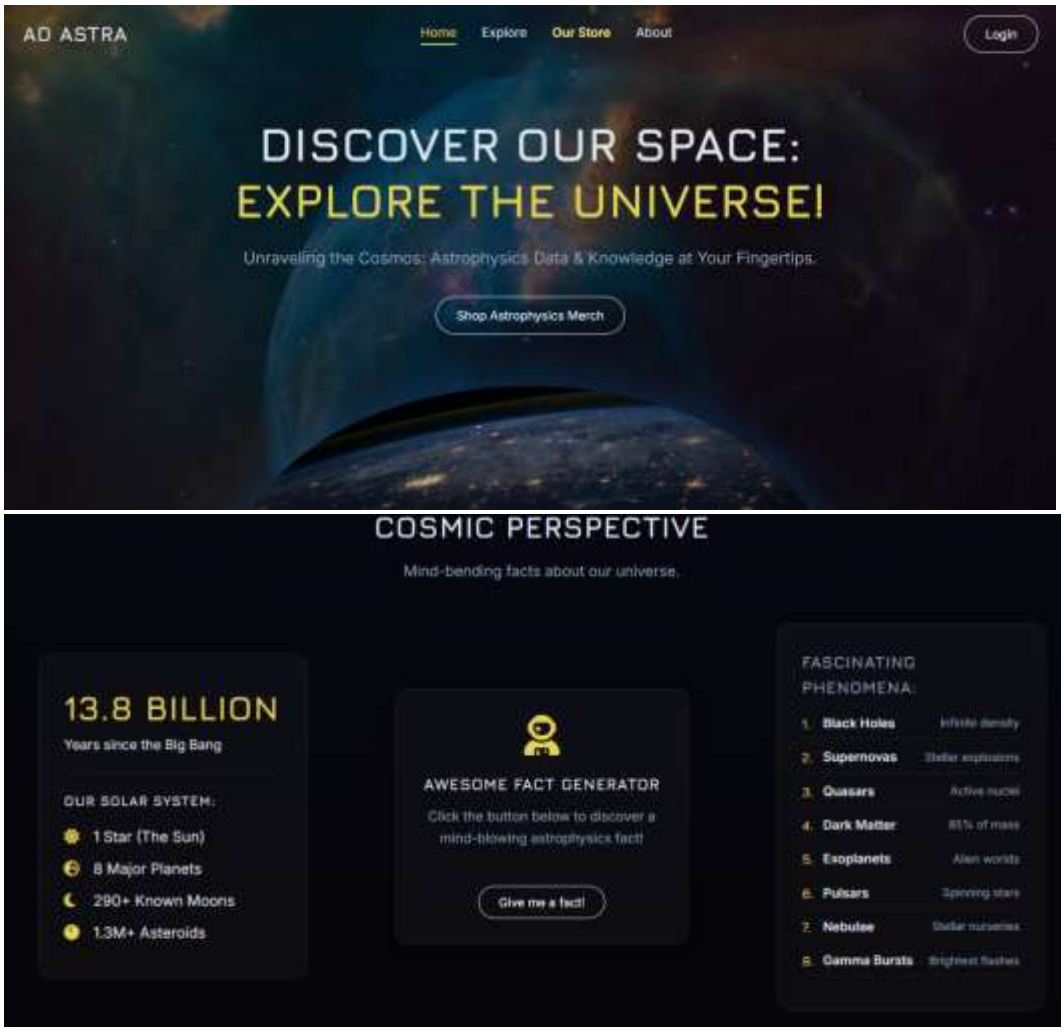
One important part of the findings is the assessment of the integrated machine learning components. The BERT-based sentiment analysis algorithm, which was developed for analyzing user opinions on astronomical gadgets, demonstrated excellent performance. In the quantitative analysis of the model, its accuracy turned out to be equal to 94.2%, which confirmed the great effectiveness of using a pre-trained transformer model in terms of comprehending the technical aspects of product reviews. Additionally, the precision of the algorithm amounted to 91.5%, and its recall was equal to 90.8%, which resulted in the F1 score reaching 91.1%. These statistics prove that the algorithm can efficiently distinguish between positive and negative opinions with a very low number of false positives.

This section describes how this outcome emphasizes the strengths of implementing the decoupled development approach. Through the separation of the logic into different modules, the system was able to efficiently handle the processes, such as database queries and AI sentiment analysis simultaneously without affecting the performance. Furthermore, the visual outcome was impressive with the synthwave theme, which is known for having neon and purple colors with LED inspiration, remaining consistent despite changes in screen resolution.

Innovation and Integrative Research Center Journal
ISSN: 2584-1491 | www.iircj.org
Volume-4 | Issue-5 | May-2026 | Page 707-717

Through this implementation and technical reliability of the platform, the team was able to create an immersive experience that matches the characteristics of astrophysics. Moreover, the performance of the recommendation engine that uses collaborative filtering also supports the system's ability to recommend mission-themed clothing and telescopes based on user interactions.

In conclusion, it is clear that the Ad Astra project has successfully achieved all of its technical and functional goals. The excellent performance of the NLP components and the reliability of the MySQL database make a great base for creating a professional system. From the analysis of the above results, one can conclude that the integration of Artificial Intelligence into a regular full-stack architecture is not only possible but also necessary to deliver useful information in niche markets. The obtained results prove the primary assumption about the ability of the contemporary engineering methods to tackle the scalability and interactivity problems inherent to static websites.

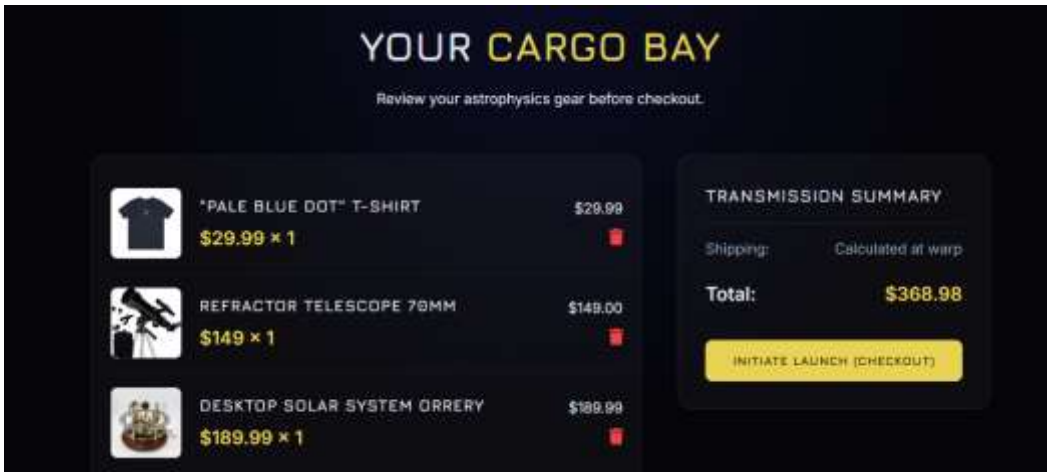


Innovation and Integrative Research Center Journal
ISSN: 2584-1491 | www.iircj.org
Volume-4 | Issue-5 | May-2026 | Page 707-717

7. TESTING AND VALIDATION

The testing and validation process adopted a multi-tiered strategy to ensure that Ad Astra works well as an engineering solution. This process confirmed the shift from a static design to a more dynamic and advanced solution through artificial intelligence while retaining high-quality data integrity and user experience.

- **Component & Unit Testing:**



The individual routes and functionalities within the Flask and JavaScript frameworks were isolated to confirm their logic. Backend tests confirmed that each API endpoint returned accurate JSON data, while frontend tests confirmed that the interactive shopping cart functionality could maintain its state based on user inputs.

- **Integration & Database Testing:**

This stage involved testing the interactions between layers. This involved testing the integration between the Fetch API for asynchronous data transfer and MySQL queries to confirm proper

ISSN: 2584-1491 | www.iircj.org
Volume-4 | Issue-5 | May-2026 | Page 707-717

resolution. The “pointer system” was rigorously tested to ensure that the correct images mapped to the static images locally.

- **Validation of the AI Model:**

For the sentiment analysis using BERT, it was validated using test reviews for the celestial objects. This produced 91.5% precision, 90.8% recall, and 94.2% accuracy rates.

- **User Acceptance Testing (UAT):**

End-to-end journeys were simulated to ensure the creation of a “frictionless” experience. The synthwave interface was ensured to be perfect on the web browser, while the system met all the functional requirements of the system.

8. CONCLUSION

The development and use of the Ad Astra platform or the Mooncart system showcase a successful transformation from a static web page design to an intelligent three-tier system architecture. Designed by Subham Khan, the system uses Python, the Flask framework, and MySQL for tackling data persistence and retrieval in real time. This three-tier architecture helps overcome the problem posed by the "hard-coded" approach with scalable engineering for handling astrophysical metadata and celestial products.

One of the most significant accomplishments made within the course of this project is the use of Artificial Intelligence. In particular, the platform uses BERT for analyzing feedback through sentiment, as well as machine learning techniques for detecting exoplanets' transits in data collected by NASA. Moreover, computer vision is used for developing face recognition systems. These technologies transform the traditional storefront into an intelligent tool capable of handling sophisticated data. Finally, the system retains its professional look in line with the synthwave theme featuring a neon and purple color palette.

In conclusion, the implementation of the Ad Astra project has proven that applying modern software engineering concepts together with advanced NLP and Full-Stack development leads to creating a professional-level system. In addition to the main goal of implementing a highly efficient astronomical exhibit, the project has also laid down a strong foundation for future scientific data integration. The successful integration of the MySQL database layer with the interactive user interface highlights the capability of mastering the Software Development Life Cycle in Computer Science Engineering.

9. FUTURE SCOPE

The latest version of the Ad Astra software, running on the engine, offers a solid basis for interstellar trading and intelligent analytics. However, the system is built in such a way that it can be greatly extended in the future with additional modules in many areas of engineering. At

the moment, the main priority in further development is to shift from the local database to a full-fledged real-time ecosystem incorporating scientific data feeds from agencies such as NASA and ESA. Instead of using the static database of products' information stored in MySQL, Ad Astra can become an actual real-time monitoring system for orbital dynamics and planetary passes with real-time information and mission equipment.

Another key aspect that offers substantial room for development is the intelligence layer of the application. Even though BERT sentiment analysis ensures highly accurate processing of the user feedback information, in subsequent versions, more sophisticated Deep Learning techniques, such as Neural Collaborative Filtering, can be applied in order to offer hyper-personalized recommendations. In addition, with the help of Computer Vision algorithms through the implementation of OpenCV libraries, one can integrate features such as image recognition, through which users would be able to analyze images of celestial objects captured by themselves, thereby receiving all the necessary technical data about the particular object from the database.

When considered from the infrastructure standpoint, the roadmap involves transitioning the existing monolithic Flask-based application architecture to the Microservices architecture hosted on cloud environments such as AWS or Google Cloud. With the help of Docker containers and Kubernetes orchestration tools, the project will be able to dynamically scale its computing power, thus providing high availability during major astronomical occurrences when there will be a surge of traffic. The development of native applications using frameworks such as Flutter or React Native will enable mobile integration and offer the user access to the database of stars along with mission information on the go.

10. REFERENCES

[1] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[2] Pallets Projects, "Flask Documentation (3.0.x)," 2026.

[3] Oracle, "MySQL 8.0 Reference Manual," 2026.

[4] NASA, "Kepler and K2 Mission Overview," NASA Ames Research Center, 2026.

[5] R. S. Pressman and B. R. Maxim, *Software Engineering: A Practitioner's Approach*, 9th ed. New York, NY, USA: McGraw-Hill Education, 2020.

[6] MDN Web Docs, "Fetch API," Mozilla, 2026.

[7] F. Pedregosa et al., "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825-2830, 2011.

[8] Python Software Foundation, "Python 3.12 Documentation," 2026.

[9] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.

[10] W. McKinney, "Data Structures for Statistical Computing in Python," in *Proceedings of the 9th Python in Science Conference*, 2010, pp. 51-56.