



**ITEJ**  
Information Technology Engineering Journals  
eISSN : 2548-2157

**ITEJ** Information  
Technology  
Engineering  
Journals

Url: <https://syekhnurjati.ac.id/journal/index.php/itej>  
Email: [itej@syekhnurjati.ac.id](mailto:itej@syekhnurjati.ac.id)

## Development of an Operating System Supporting Intelligent Predictions and Recommendations

Rakhmadi Rahman  
Information Systems  
Bacharuddin Jusuf Habibie  
Institute of Technology  
Parepare  
[rakhmadi.rahman@ith.ac.id](mailto:rakhmadi.rahman@ith.ac.id)

Alya Wulan Apriliyani  
Information Systems  
Bacharuddin Jusuf Habibie  
Institute of Technology  
Parepare  
[alyawulanapriliyani0404@gmail.com](mailto:alyawulanapriliyani0404@gmail.com)

Siti Nur Azizah Ibrahim  
Information Systems  
Bacharuddin Jusuf Habibie  
Institute of Technology  
Parepare  
[sitinurazizah@gmail.com](mailto:sitinurazizah@gmail.com)

**Abstract**— This Study discusses the development of an intelligent operating system feature that supports smart predictions and recommendations using artificial intelligence (AI) capabilities within the Linux operating system. The study aims to integrate AI-driven features into Linux to enhance user productivity and efficiency by providing relevant application recommendations based on user behavior patterns. The implementation involves data collection of application usage, training machine learning models for application recommendations, and integrating these features into the Linux environment. The project utilizes Python for scripting, employing libraries such as psutil, pandas, scikit-learn, and joblib for data handling and machine learning tasks. The results demonstrate successful implementation of the AI-driven recommendation system, enhancing user interaction and productivity within the Linux operating system.

**Keywords**— AI, Linux, application recommendation

### I. INTRODUCTION

In this modern era, operating systems have become the core of our daily computing experience. Linux, as one of the most widely used operating systems in the world, continues to evolve rapidly to meet the increasingly complex needs of its users [1]. One of the main challenges users face is how to increase productivity and efficiency in using the applications available within this operating system [2][3]. Although Linux offers high flexibility and control, users often encounter difficulties in finding and using the applications that best suit their needs[4].

This research aims to address these challenges by developing an intelligent operating system capable of providing application recommendations based on individual usage patterns and preferences. The system will use artificial intelligence (AI) technology to analyze application usage data and user behavior patterns [5][6]. Thus, it is expected that

this system can help Linux users choose and use applications more efficiently, according to their daily needs.

The main problem this research seeks to solve is the lack of effective mechanisms to recommend applications to Linux users based on their usage habits[7]. Currently, users often have to rely on manual searches or recommendations from external sources to find applications that meet their needs. This can result in wasted time and suboptimal application use. To address this issue, the proposed approach is the development of an AI-based recommendation system that is directly integrated into the Linux kernel. This system will utilize distributed application usage log data throughout the system to identify usage patterns. Subsequently, based on this analysis, the system will provide application recommendations that best match the user's preferences and needs.

Relevant literature shows that the use of artificial intelligence technology in operating systems for application recommendation purposes has become a significant focus of research in recent years. Various approaches have been proposed, ranging from the use of collaborative filtering algorithms to deep learning for analyzing usage patterns. However, this research will consider the unique aspects related to the Linux ecosystem and optimal implementation within it. It is hoped that this research will result in an intelligent operating system that not only enhances the Linux user experience in using applications but also opens up new potential for integrating artificial intelligence technology into open-source operating systems. The benefits of this research include increased user productivity, reduced time wasted in searching for applications, and overall optimization of system resource use. Thus, this research is expected to make a significant contribution to the development of a more adaptive and responsive Linux operating system that meets the needs of modern users.

## **II. LITERATURE REVIEW**

### **A. Operating Systems and Artificial Intelligence (AI)**

An operating system[8], at its core, is software that regulates and manages various aspects of computer hardware. Since the advent of modern computing, operating systems have become the backbone of daily computer use. Their primary functions include resource allocation such as memory and CPU, file management, and providing an interface that allows users to interact with the computer.

An operating system is a collection of software routines that exist between application programs and hardware. All software runs under the control of the operating system, accesses the hardware via the operating system, and follows the rules enforced by the operating system. The operating system functions like a government in a country, in the sense that it creates conditions for computers to be able to run programs correctly[9]. The operating system is the most important program of the programs contained in a computer system. The operating system can be considered as a control program whose job is to run other programs on the computer. An OS (Operating System) extends the machine and gives programmers a simpler way to work with the hardware.

The operating system has systematic scheduling including calculating memory usage, processing data, data storage, and other resources. For hardware functions such as input and output and memory allocation, the operating system acts as an intermediary between application programs and computer hardware, although application code is usually executed directly by the hardware and will often contact the OS or be interrupted by it. The operating system is found on almost any device that contains a computer—from cell phones and video game consoles to supercomputers and web servers. Examples of modern operating systems are Linux, Android, iOS, Mac OS, Microsoft Windows[10]. The position of the operating system is between applications and hardware .

The integration of artificial intelligence (AI) into operating systems expands the role of the operating system from merely managing resources to becoming an entity capable

of understanding and intelligently responding to user usage patterns [11][12]. AI enables the operating system to predict user needs based on behavior and habits, and to learn from experience to provide more personalized and relevant recommendations.

AI is always the frontier of computational advances that address ever more complex decision-making problems. McCorduck (2004) suggests AI is made up of the “tough nuts,” the next step in computing that computer scientists are ironing out[13]. Some even consider AI to be simply “what AI researchers do” at any point in time, knowing that AI is always necessarily at the edge of our current capabilities and a moving target. Once we are using it in practice, it is no longer AI. At the present time, we often reserve the term AI for machine learning algorithms based on neural networks and data driven predictions. But this view will no longer be satisfactory when the current class of machine learning algorithms gives way to the next generation.

Artificial Intelligence (AI) is a specialized branch of computer science focused on tackling cognitive challenges that are typically linked to human intelligence, such as learning, image creation, and recognition [14]. This field encompasses a diverse array of techniques and methodologies, including machine learning, natural language processing (NLP), artificial neural networks, and genetic algorithms. Machine learning involves training algorithms to improve their performance on tasks through experience, while NLP enables computers to understand and generate human language. Artificial neural networks, inspired by the human brain's structure, are used to recognize patterns and make decisions, and genetic algorithms mimic the process of natural evolution to optimize solutions. By leveraging these sophisticated algorithms and mathematical models, AI can process and analyze vast quantities of data, uncovering patterns that are often beyond human perception. This capability allows AI systems to make accurate predictions and offer valuable recommendations based on their analysis. The applications of AI are widespread and impactful across various industries. In healthcare[15], AI assists in diagnosing diseases and personalizing treatments; in finance, it enhances fraud detection and risk management; in transportation, it improves traffic management and supports the development of autonomous vehicles; and in agriculture, AI optimizes crop management and boosts yield. These advancements contribute significantly to increasing efficiency and reducing costs, demonstrating the transformative potential of AI technology.

## **B. Intelligent Predictions and Recommendations in the Context of Operating Systems**

The concept of prediction in the context of operating systems involves the system's capability to forecast the applications or services that users are likely to require based on their historical interactions. This predictive functionality leverages data such as the time of day, the user's location, and the activities that the user frequently engages in to make informed guesses about what applications the user might want to access next. For instance, if a user typically opens their email application every morning at 8 AM, the operating system might preemptively load the email application around that time to enhance the user experience.

Intelligent recommendations take this a step further by employing machine learning techniques to analyze comprehensive datasets gathered from previous user behavior [16]. These techniques involve identifying patterns and trends in how and when users interact with different applications. By comprehending these usage patterns, the system can generate tailored recommendations that are highly relevant to the user. For example, if the system notices that a user frequently switches between a web browser and a document editor during work hours, it might suggest these applications during that time period. Additionally, the system can suggest actions that align with the user's past behavior, such as recommending a specific document or website they often visit at a certain time. This intelligent recommendation process not only streamlines the user's workflow but also

makes the operating system more intuitive and personalized, enhancing overall user satisfaction and productivity.

In the current era of digital transformation, integration between business information systems and machine learning technology is becoming increasingly important to increase company efficiency and competitiveness. Business information systems (SIB) are the main foundation that supports data management processes and decision making in various business fields. On the other hand, machine learning technology has opened up new opportunities with its ability to process data automatically and produce more accurate predictions based on identified patterns.

The main components in Artificial Intelligence (AI) include machine learning, natural language processing, computer vision, logical reasoning, knowledge representation, decision making, robotics and control, context understanding and pattern recognition, intelligent agent development, and integration and interoperability. These components are interrelated and support each other in the development and application of AI.

### **C. Implementation of AI Smart Features in the Linux Operating System**

A case study on the implementation of AI smart features in the Linux operating system highlights various projects and research efforts that integrate AI technology into different aspects of the system. These initiatives demonstrate how machine learning methods, such as clustering for user segmentation and classification for pattern recognition, are utilized to enhance the prediction and recommendation capabilities within the Linux operating system. Clustering helps group users based on similar behavior or preferences, enabling more personalized experiences, while classification identifies patterns in user data to make accurate predictions about future actions.

The development of applications and algorithms in the Linux context involves the application of machine learning techniques optimized for an open-source environment. This is facilitated by powerful tools such as TensorFlow, scikit-learn, and PyTorch, which support the efficient development and implementation of AI models. TensorFlow is widely used for building and training neural networks, scikit-learn offers simple and efficient tools for data mining and data analysis, and PyTorch provides flexibility and speed in developing deep learning models. These tools enable developers to create robust AI applications that can be seamlessly integrated into the Linux operating system[17].

Linux, a Unix-like operating system, was designed to offer PC users a free or low-cost alternative comparable to traditional and more expensive Unix systems. Its open-source nature allows for extensive customization and development, making it an ideal platform for integrating advanced AI technologies. The integration of AI into Linux not only leverages the system's flexibility and robustness but also enhances its functionality by providing smarter and more adaptive user experiences. This makes Linux a powerful and versatile operating system capable of meeting the evolving needs of users in various sectors [18].

### **D. Challenges and Solutions in Implementation**

The main challenges in integrating artificial intelligence into operating systems include managing large amounts of user data, complex privacy regulations, and ensuring optimal system performance without compromising security. Solutions to these challenges involve developing efficient algorithms, optimizing resource usage, and implementing strict privacy policies to protect user data[19].

The main challenges in integrating artificial intelligence in operating systems include managing large amounts of user data, complex privacy settings[20], and ensuring optimal system performance without compromising security. Here are some specific challenges and proposed solutions for each:

1. User Data Management

**Challenge:** Collecting and storing large amounts of application usage data can be a heavy burden on the system, both in terms of storage space and processing performance.

**Solution:** Implementation of data compression techniques and distributed storage can help reduce storage load. In addition, using efficient and parallel data processing algorithms can increase data processing speed.

2. User Privacy

**Challenge:** Managing users' personal data requires special attention to privacy and security. The risk of data leaks and unauthorized use of data is a significant problem.

**Solution:** Implement strict data encryption and security protocols to protect user information. Additionally, providing users with options to control and manage their personal data, as well as transparency in data usage, will increase user trust.

3. System Performance

**Challenge:** Running complex artificial intelligence algorithms can impact overall operating system performance, especially on devices with low specifications.

**Solution:** Optimize AI algorithms and models for performance efficiency, and utilize hardware acceleration such as GPU or TPU. Additionally, techniques such as model distillation can be used to reduce model size without sacrificing accuracy.

4. Integration with Existing Systems

**Challenge:** Integrating AI smart features with existing operating systems can create conflicts with existing applications and services.

**Solution:** Conduct extensive testing on various system and hardware configurations to ensure compatibility and stability. The use of containerization (eg, Docker) can help isolate newly integrated applications and services.

5. Resource Availability

**Challenge:** Developing and maintaining AI intelligent features requires significant resources, including expertise, time, and money.

**Solution:** Adopting an open-source and collaborative approach to development can reduce the burden on resources. Collaborating with the open-source developer community can make significant contributions in terms of feature development, maintenance, and enhancement.

By facing and overcoming these challenges, the integration of artificial intelligence in the Linux operating system is expected to provide major benefits to users, increase productivity, and unlock new potential for the development of AI technology in everyday computing environments.

## E. Benefits and Future Development Potential

The use of operating systems that support intelligent predictions and recommendations has great potential to increase user productivity and provide a more personalized user experience. By leveraging increasingly sophisticated AI technology, such as using sensor data for real-time predictions and adapting to smart devices, operating systems can become more responsive and adaptive to the needs of modern users.

### Benefits

1. Increased User Productivity

- **Relevant Application Recommendations:** By providing application recommendations based on user usage patterns and preferences, the system can help users find the most effective and efficient tools for their needs, thereby enhancing overall productivity.
- **Time Savings:** Reducing the time spent searching for the right application allows users to focus on more important tasks.

2. Personalized User Experience

- **Adaptive Interface:** An AI-equipped operating system can adjust its interface and functions based on user habits, offering a more personalized and responsive experience.
- **Predicting Needs:** The system's predictive capabilities can assist users by suggesting applications or actions before they realize they need them, making interaction with the computer smoother and more intuitive.
- 3. **More Efficient Resource Usage**
  - **Resource Optimization:** By understanding application usage patterns, the system can allocate resources such as CPU and memory more efficiently, reducing waste and enhancing overall system performance.
  - **Reduced Manual Workload:** Automation in application recommendations can reduce the need for manual adjustments by users, thereby reducing workload and potential errors.

#### **Future Development Potential**

1. **Integration with Other Technologies**
  - **IoT and Smart Devices:** Integration with IoT devices can extend the predictive and recommendation capabilities to various devices, creating a more connected and intelligent ecosystem.
  - **Virtual and Augmented Reality:** Using VR and AR technologies can create more immersive and interactive computing experiences, where application recommendations are tailored to the context of use within virtual or augmented environments.
2. **Enhanced AI Algorithms**
  - **Advanced Deep Learning Models:** Adopting more sophisticated deep learning algorithms can improve the accuracy and effectiveness of predictions and recommendations, providing more relevant and reliable results.
  - **Adaptive Learning:** Developing AI models that continually learn and adapt to changes in user behavior over time ensures that recommendations remain relevant and up-to-date.
3. **User Interface Development**
  - **More Interactive UI/UX Design:** Developing more interactive and intuitive interfaces can enhance the user experience when interacting with the recommendation system.
  - **Assistive Interfaces:** Implementing interfaces that support various user needs, including those with disabilities, can make this technology more inclusive and accessible to a broader audience.
4. **Expansion to Other Platforms**
  - **Multi-Platform Support:** Developing recommendation capabilities that can be applied across different operating systems and platforms, such as Windows, macOS, or Android, to extend the benefits and applications of this technology to more users.
  - **Cloud Integration:** Integrating the recommendation system with cloud services can improve data processing and storage capabilities, as well as enable cross-device synchronization.
5. **Enhanced Security and Privacy**
  - **Implementation of Latest Privacy Technologies:** Adopting the latest technologies in data encryption and anonymization to enhance user security and privacy.
  - **Development of Transparent Privacy Policies:** Increasing transparency in data usage and providing users with full control over their personal data can increase trust and adoption of this technology.

With various benefits and future development potential, integrating intelligent AI features into the Linux operating system not only provides a better user experience but also paves the way for more advanced and responsive technology innovations to meet modern user needs.

### III. METHOD

This research method is designed to develop an AI-based application recommendation system on the Linux operating system[21]. Below are the details regarding the research design, population and sample, data collection techniques, and data analysis techniques used. This study uses an experimental approach to develop and test the AI-based application recommendation system. The experimental approach was chosen to allow controlled testing of the effectiveness of the developed recommendation system.

The research population consists of active Linux operating system users within the open-source community. The sample used was purposively selected from a community of users who have a deep understanding of the Linux operating system and their application needs.

#### 1. Data Collection Techniques and Development Instruments:

Data is collected through the analysis of application usage logs. Application logging is used to record user interactions with specific applications on the Linux operating system. This data includes information about the frequency of use, time of use, and types of applications used

#### 2. The instruments used include:

Application Logging: Recording application usage data to analyze usage patterns that can be used as a basis for developing the recommendation model. Data from application usage logs are analyzed using AI techniques, such as classification and clustering-based modeling. This analysis aims to identify usage patterns that can be used as the foundation for the application recommendation system.

The software specifications used for system development include a Python development environment with key libraries such as TensorFlow for AI models and Flask for web application development. The data collected and processed with this system is expected to provide valuable insights for further development and enhancement of the Linux operating system in terms of more efficient and effective application recommendations.

### IV. RESULTS AND DISCUSSION

The results of this study describe the development and evaluation of an AI-based application recommendation system on the Linux operating system. The main findings of this research are as follows: Figure 1. Figure caption

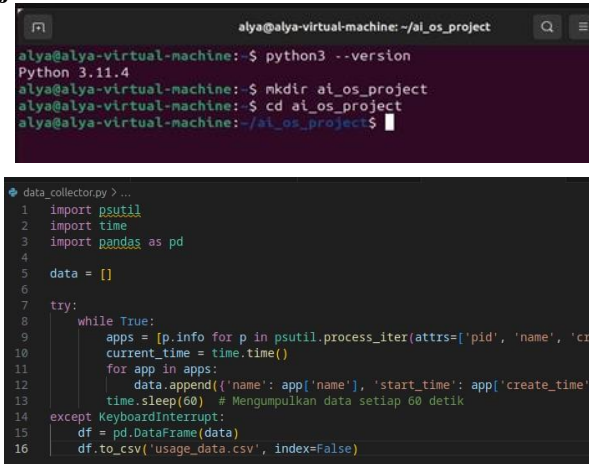
#### 1. Installing Python and Using It in Visual Studio Code

Python plays a crucial role in this project as the programming language used for:

- a) Data Collection:  
Use the psutil library to monitor running processes on the system and record the start time and duration of application usage.
- b) Data Processing:  
Using the pandas library to store and process data in the form of a DataFrame.
- c) Training the Machine Learning Model:  
Using the scikit-learn library to create and train an application prediction model based on the collected data.
- d) Providing Recommendations:

Using the trained model to provide real-time application recommendations through system notifications.

## 2. Creating a Project Folder in the Terminal



```

alya@alya-virtual-machine: ~/ai_os_project
alya@alya-virtual-machine:~$ python3 --version
Python 3.11.4
alya@alya-virtual-machine:~$ mkdir ai_os_project
alya@alya-virtual-machine:~$ cd ai_os_project
alya@alya-virtual-machine:~/ai_os_project$

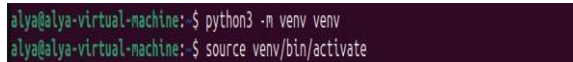
data_collector.py > ...
1 import psutil
2 import time
3 import pandas as pd
4
5 data = []
6
7 try:
8     while True:
9         apps = [p.info for p in psutil.process_iter(attrs=['pid', 'name', 'create_time'])]
10        current_time = time.time()
11        for app in apps:
12            data.append({'name': app['name'], 'start_time': app['create_time'], 'current_time': current_time})
13        time.sleep(60) # Mengumpulkan data setiap 60 detik
14    except KeyboardInterrupt:
15        df = pd.DataFrame(data)
16        df.to_csv('usage_data.csv', index=False)

```

Figure 1. Running Python and Creating

Running Python and Creating a Project Folder First, run Python and create the project folder. The directory `ai_os_project` is created and used as the working directory for all project files. After creating the folder, navigate to Visual Studio Code.

## 3. Setting Up a Virtual Environment in the Terminal



```

alya@alya-virtual-machine:~$ python3 -m venv venv
alya@alya-virtual-machine:~$ source venv/bin/activate

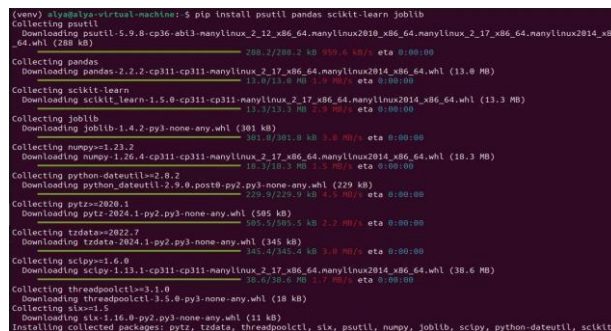
```

Figure 2. Setting Up a Virtual Environment in the Terminal

Creating a Virtual Environment: In the terminal inside the project folder, create a virtual environment and Activate Virtual Environment . The virtual environment is created and activated. This environment will be used to isolate project dependencies from the main Python system.

## 4. Install Dependencies

In the VS Code terminal inside the project folder, install the required libraries with the command:



```

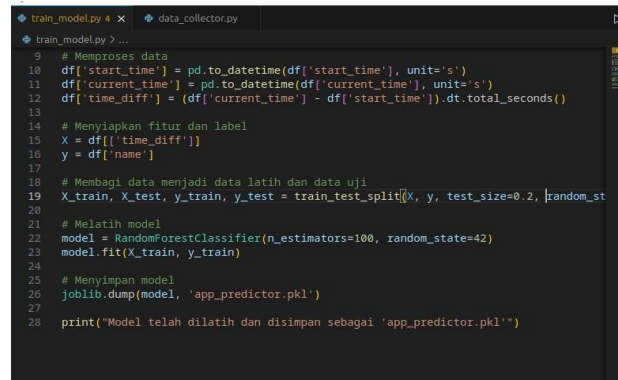
(venv) alya@alya-virtual-machine:~/ai_os_project$ pip install psutil pandas scikit-learn joblib
Collecting psutil
  Downloading psutil-5.9.8-cp36-abi3-manylinux_2_12_x86_64.manylinux2010_x86_64.manylinux2014_x86_64.whl (288 kB)
Collecting pandas
  Downloading pandas-2.2.2-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (13.0 MB)
Collecting scikit-learn
  Downloading scikit-learn-1.5.0-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (13.3 MB)
Collecting joblib
  Downloading joblib-1.4.2-py3-none-any.whl (301 kB)
Collecting numpy<=1.23.2
  Downloading numpy-1.26.4-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (18.3 MB)
Collecting python-dateutil<=2.8.2
  Downloading python_dateutil-2.9.0.post0-py3-none-any.whl (229 kB)
Collecting pytz<=2020.1
  Downloading pytz-2020.1-py2.py3-none-any.whl (505 kB)
Collecting tzdata<=2022.7
  Downloading tzdata-2024.1-py2.py3-none-any.whl (345 kB)
Collecting scipy<=1.6.0
  Downloading scipy-1.13.1-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (38.6 MB)
Collecting threadpoolctl<=3.1.0
  Downloading threadpoolctl-3.5.0-py3-none-any.whl (18 kB)
Collecting six<=1.5
  Downloading six-1.16.0-py2.py3-none-any.whl (11 kB)
Installing collected packages: pytz, tzdata, threadpoolctl, six, psutil, numpy, joblib, scipy, python-dateutil, scikit-learn, pandas

```

Figure 3. Install Dependencies

The libraries installed in the Visual Studio Code (VS Code) terminal in the project folder aim to provide the functional support required by the application being developed[22]. In the context of software development, the use of libraries is crucial because they are provided





```

train_model.py 4 x data_collector.py
train_model.py > ...
9 # Memproses data
10 df['start_time'] = pd.to_datetime(df['start_time'], unit='s')
11 df['current_time'] = pd.to_datetime(df['current_time'], unit='s')
12 df['time_diff'] = (df['current_time'] - df['start_time']).dt.total_seconds()
13
14 # Menyiapkan fitur dan label
15 X = df[['time_diff']]
16 y = df['name']
17
18 # Membagi data menjadi data latih dan data uji
19 X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_st
20
21 # Melatih model
22 model = RandomForestClassifier(n_estimators=100, random_state=42)
23 model.fit(X_train, y_train)
24
25 # Menyimpan model
26 joblib.dump(model, 'app_predictor.pkl')
27
28 print("Model telah dilatih dan disimpan sebagai 'app_predictor.pkl'")

```

Figure 4. Module

Ready-to-use modules that can be used for various purposes such as data processing, creating machine learning models, visualization, and interaction with operating systems. For example, in the application you are developing, libraries such as psutil are used to access information about the processes running in the operating system, while pandas is used for manipulation and analysis of data in the form of dataframes. Meanwhile, scikit-learn and joblib are very useful for building and training machine learning models, as well as saving these models in a form that can be recalled.

Installation of this library is carried out in a virtual environment (venv) to ensure that the application being developed has an isolated and clean working environment. This helps prevent conflicts between different library versions and makes overall application dependency management easier.

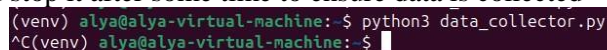
## 5. Usage Data Collection Script

Data\_collector.py serves as the first part of the data pipeline in your program. Its main function is to collect data about applications running on the system and save it into a CSV file (usage\_data.csv). This data includes information such as the application name, start time, and the time when the data was collected.

- Application Usage Data Collection:
- This script collects data about applications running on the system, including process ID ( pid ), application name (name), and application start time (create\_time).
- Data collection was carried out repeatedly with a time delay of 60 seconds between each data collection.
- The collected data is stored in a data list, which is then converted into a DataFrame using pandas.
- When the script is stopped (for example, by pressing Ctrl+C), the collected data is saved into a CSV file named usage\_data.csv.
- The data stored in usage\_data.csv is used as a dataset to train the machine learning model.
- This model is then used to predict and provide application recommendations based on usage patterns found in the data.
- By running these scripts automatically (for example, using systemd services), application usage data can be collected continuously without manual intervention.

The data\_collector.py program is very useful for collecting application usage data on a single computer and can be used for local analysis and for training machine learning models. However, to know the overall number of Linux users, a different and broader approach and infrastructure is needed.

Run the script and stop it after some time to ensure data is collected



```

(venv) alya@alya-virtual-machine:~$ python3 data_collector.py
^C(venv) alya@alya-virtual-machine:~$

```

Figure 5. Usage Data Collection Script

## 6. Machine Learning Model Training Thesis

Accuracy is calculated by comparing the model predictions ( $y_{pred}$ ) with the actual labels ( $y_{test}$ ). Accuracy results are printed to give an idea of how well the model is working. A model training script was written to read data from `usage_data.csv`, train a KNearest Neighbors model, and save the model to `app_predictor.pkl`.

```
^C(venv) alya@alya-virtual-machine:~/ai_os_project$ python train_model.py
Accuracy: 0.06153846153846154
```

Figure 6. Runs the Application model training script

The accuracy function in the context of machine learning is a metric used to measure how well a prediction model performs. Accuracy measures the proportion of correct predictions compared to the total number of predictions made.

## 7. Application Recommendation Service Script

```
app_recommender.py > ...
1 import pandas as pd
2 import psutil
3 import time
4 import joblib
5 import os
6
7 # Load the trained model
8 model = joblib.load('app_predictor.pkl')
9
10 # Infinite loop to continuously monitor and recommend apps
11 try:
12     while True:
13         # Get the list of current running applications
14         apps = [p.info for p in psutil.process_iter(attrs=['pid', 'name', 'create_
15             current_time = time.time()
16
17         for app in apps:
18             duration = current_time - app['create_time']
19             # Create a DataFrame with the same structure as used during training
20             X = pd.DataFrame([duration], columns=['duration'])
21             # Predict the application name
22             recommended_app = model.predict(X)[0]
23             # Display recommendation
24             os.system(f'notify-send "Recommendation" "Consider using {recommended_
25
26         time.sleep(60) # Run every 60 seconds
27
28 except KeyboardInterrupt:
29     print("Recommendation service stopp
30
31 Do you want to install the recommended 'Rainbow CSV'
32 extension from mechatroner for usage_data.csv?
```

Figure 7. Application Recommendation Service Script

Function of the `app_recommender.py` Code

1. **Application Monitoring:** This code monitors the applications running on the system.
2. **Predictions and Recommendations:** Based on the duration of current app usage, this code predicts other apps that may be more relevant or useful for the user using a pre-trained model.
3. **User Notifications:** This code sends notifications to users with app recommendations, helping to increase productivity or provide better suggestions.

Recommendation service scripts are written to load the trained model and provide application recommendations based on current time input and usage duration. Recommendation service scripts are written to load the trained model and provide application recommendations based on current time input and usage duration.

Running Recommendation Scripts

```
(venv) alya@alya-virtual-machine:~/ai_os_project$ python app_recommender.py
=====
RECOMMENDATION SYSTEM
=====
Consider using: rcu_gp
=====
```

Figure 8. Running Recommendation Scripts

This script will run in the background and provide app recommendations based on the predictions of the trained model.

## 8. User Interface (User Interface)

### a) Home Page

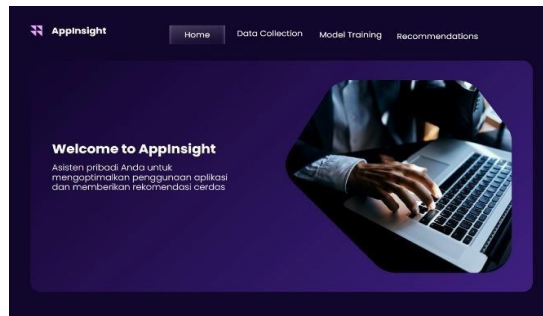


Figure 9. Home Page

On this main page, the goal is to provide a brief introduction and welcome users into the application. The page header displays the app's logo and name to build a strong visual identity. The navigation located at the top allows users to easily move between the main pages of the application. The center of the main page provides a clear and user-friendly information box, providing a brief guide to what users can do with the app

### b) Data Collection Page

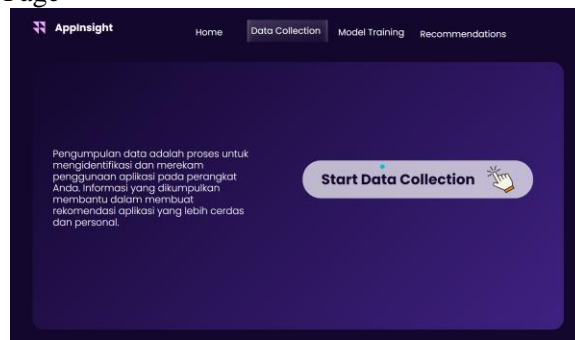


Figure 10. Data Collection Page

This page is designed with a focus on the data collection process which is an important first step in the functioning of this application. A large, clear button is provided to start data collection, emphasizing the action the user must take.

### c) Model Training Page

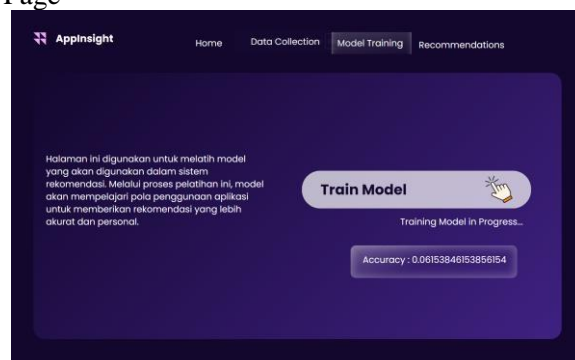


Figure 11. Model Training Page

On this page, the main focus is to provide information about the model training process in a visual and easy to understand way. The design displays visual

components such as a progress bar, and once completed it will display the application's accuracy results based on the duration of its use

d) Recommendations Page

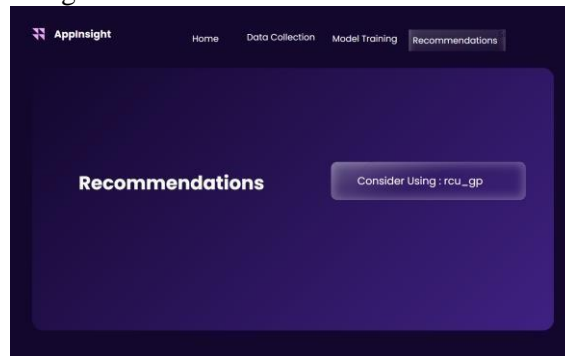


Figure 12. Recommendations Page

This page is designed to provide app recommendations based on the predictions of a previously trained model. The design emphasizes visual appropriateness and clarity of information to ensure users can easily understand the recommendations provided. Recommendation information is presented in a structured and intuitive format, perhaps in the form of a box containing application recommendation text

## V. CONCLUSION

This study aims to develop a Linux-based operating system equipped with intelligent AI features for application prediction and recommendation. At the implementation stage, the development process includes installing Visual Studio Code and Python extensions that support application development, as well as creating and setting up project folders within the VS Code environment. The use of the Python programming language as the backbone of the project allows collecting application data, processing data using pandas, training machine learning models with scikit-learn, and implementing recommendation services based on the trained models. The data\_collector.py script is used to collect application usage data and save it in CSV format, which is the first step in the model building process. The train\_model.py script is responsible for training the application prediction model using the collected data, using the KNearest Neighbors (KNN) technique to produce a model that can provide recommendations based on application usage patterns. The implementation of the application recommendation service is realized in the app\_recommender.py script, where the trained model is taken from the app\_predictor.pkl file to provide real-time application recommendations based on current usage data. Users are provided with notifications regarding recommended applications, with the aim of increasing efficiency and user experience in using this intelligent operating system. The integration of intelligent AI features in the Linux operating system provides prediction and recommendation capabilities that can increase user productivity, enabling more optimal use of applications based on identified usage patterns. Thus, the development of this operating system not only opens up opportunities for more effective use of applications, but also shows the potential application of AI technology in the context of managing and providing recommendations in everyday computing environments.

## REFERENCES

- [1] J. V. Souto dan M. Castro, "Improving concurrency and memory usage in distributed operating systems for lightweight manycores via cooperative time-sharing lightweight tasks," *J. Parallel Distrib. Comput.*, vol. 174, hal. 2–18, Apr 2023, doi: 10.1016/j.jpdc.2022.12.006.

- [2] A. Shoshi, R. Mieke, dan T. Bauernhansl, "Conceptual Thoughts on Biointelligent Embedded Systems and Operating Systems Architecture," *Procedia Comput. Sci.*, vol. 217, hal. 969–978, 2023, doi: 10.1016/j.procs.2022.12.294.
- [3] M. Rueben dan W. D. Smart, "A Shared Autonomy Interface for Household Devices [ Extended Abstract ]," *Proc. Tenth Annu. ACM/IEEE Int. Conf. Human-Robot Interact. Ext. Abstr.*, hal. 165–166, 2015, doi: 10.1145/2701973.2702061.
- [4] E. Freitas, A. T. de Oliveira Filho, P. R. X. do Carmo, D. Sadok, dan J. Kelner, "A survey on accelerating technologies for fast network packet processing in Linux environments," *Comput. Commun.*, vol. 196, hal. 148–166, Des 2022, doi: 10.1016/j.comcom.2022.10.003.
- [5] A. Zare *et al.*, "An update for various applications of Artificial Intelligence (AI) for detection and identification of marine environmental pollutions: A bibliometric analysis and systematic review," *Mar. Pollut. Bull.*, vol. 206, hal. 116751, Sep 2024, doi: 10.1016/j.marpolbul.2024.116751.
- [6] Z. Liu dan C. Yang, "Exploring group-level human mobility from location-based social media check-in data," *Proc. 5th IEEE Conf. Ubiquitous Positioning, Indoor Navig. Locat. Serv. UPINLBS 2018*, hal. 1–5, 2018, doi: 10.1109/UPINLBS.2018.8559796.
- [7] A. Orlando *et al.*, "Linux page fault analysis in android systems," *Microprocess. Microsyst.*, vol. 66, hal. 10–18, Apr 2019, doi: 10.1016/j.micpro.2019.01.006.
- [8] D. Chen, Y. Qiao, Y. Sun, dan X. Gao, "Human reliability assessment and risk prediction for deep submergence operating system of manned submersible under the influence of cognitive performance," *Ocean Eng.*, vol. 266, hal. 112753, Des 2022, doi: 10.1016/j.oceaneng.2022.112753.
- [9] E. Marková, P. Sokol, S. P. Krišáková, dan K. Kováčová, "Dataset of Windows operating system forensics artefacts," *Data Br.*, vol. 55, hal. 110693, Agu 2024, doi: 10.1016/j.dib.2024.110693.
- [10] M. Asim, M. F. Amjad, W. Iqbal, H. Afzal, H. Abbas, dan Y. Zhang, "AndroKit: A toolkit for forensics analysis of web browsers on android platform," *Futur. Gener. Comput. Syst.*, vol. 94, hal. 781–794, Mei 2019, doi: 10.1016/j.future.2018.08.020.
- [11] S. Isaac, M. R. Phillips, K. A. Chen, R. Carlson, C. C. Greenberg, dan S. Khairat, "Usability, Acceptability, and Implementation of Artificial Intelligence (AI) and Machine Learning (ML) Techniques in Surgical Coaching and Training: A Scoping Review," *J. Surg. Educ.*, Mei 2024, doi: 10.1016/j.jsurg.2024.03.018.
- [12] K. Wang, X. Yan, Z. Zhu, dan X. (Michael) Chen, "Understanding bike-sharing usage patterns of members and casual users: A case study in New York City," *Travel Behav. Soc.*, vol. 36, hal. 100793, Jul 2024, doi: 10.1016/j.tbs.2024.100793.
- [13] D. Rodoplu Solovchuk, "Advances in AI-assisted biochip technology for biomedicine," *Biomed. Pharmacother.*, vol. 177, hal. 116997, Agu 2024, doi: 10.1016/j.biopha.2024.116997.
- [14] S. Hadzovic, L. Becirspahic, dan S. Mrdovic, "It's time for artificial intelligence governance," *Internet of Things*, vol. 27, hal. 101292, Okt 2024, doi: 10.1016/j.iot.2024.101292.
- [15] D. Voke, A. Perry, S. H. Bardach, N. S. Kapadia, dan A. E. Barnato, "Innovation pathways to preserve: Rapid healthcare innovation and dissemination during the COVID-19 pandemic," *Healthcare*, vol. 10, no. 4, hal. 100660, Des 2022, doi: 10.1016/j.hjdsi.2022.100660.
- [16] S.-Z. Chiou-Wei dan Y.-T. Lee, "Application of KL distance-based intelligent recommendation method to fund recommendation for users with investment behavior in Asia Region," *Heliyon*, vol. 10, no. 12, hal. e32959, Jun 2024, doi: 10.1016/j.heliyon.2024.e32959.

- [17] P. Grzesik dan D. Mrozek, “Combining Machine Learning and Edge Computing: Opportunities, Challenges, Platforms, Frameworks, and Use Cases,” *Electronics*, vol. 13, no. 3, hal. 640, Feb 2024, doi: 10.3390/electronics13030640.
- [18] G. Duan, Y. Fu, M. Cai, H. Chen, dan J. Sun, “DongTing: A large-scale dataset for anomaly detection of the Linux kernel,” *J. Syst. Softw.*, vol. 203, hal. 111745, Sep 2023, doi: 10.1016/j.jss.2023.111745.
- [19] H. Reza Amri, Ridho Taufiq Subagio, dan Kusnadi, “Penerapan Metode CSI untuk Pengukuran Tingkat Kepuasan Layanan Manajemen,” *J. Sist. Cerdas*, vol. 3, no. 3, hal. 241–252, 2020, doi: 10.37396/jsc.v3i3.86.
- [20] T.-H. Lee, S. Kim, J. Lee, dan C.-H. Jun, “Word2Vec-based efficient privacy-preserving shared representation learning for federated recommendation system in a cross-device setting,” *Inf. Sci. (Ny)*, vol. 651, hal. 119728, Des 2023, doi: 10.1016/j.ins.2023.119728.
- [21] H. Wang, Z. Chen, G. Xiao, dan Z. Zheng, “Network of networks in Linux operating system,” *Phys. A Stat. Mech. its Appl.*, vol. 447, hal. 520–526, Apr 2016, doi: 10.1016/j.physa.2015.12.084.
- [22] Y. Guo *et al.*, “A framework for threat intelligence extraction and fusion,” *Comput. Secur.*, vol. 132, hal. 103371, Sep 2023, doi: 10.1016/j.cose.2023.103371.