Petroleum Physical Properties Prediction Application in Enhanced Oil Recovery Process

Harry Budiharjo Sulistyarso, Dyah Ayu Irawati, Joko Pamungkas, Indah Widiyaningsih

Abstract: The Enhanced Oil Recovery (EOR) process is one of the ways in the petroleum exploitation process so that thick oil can be lifted to the surface and produced. The EOR process referred to in this study is the EOR process carried out in previous studies at the EOR laboratory of UPN Veteran Yogyakarta Indonesia by adding biosurfactants and adjusting the temperature. In laboratory experiments, each time an amount of biosurfactant concentration is added and the temperature is adjusted, the calculation must be done repeatedly to determine the amount of viscosity, interfacial tension (IFT), and density. This experiments takes a long time, requires high cost and variety limitation of the condition. The previous research has succeeded in building a model with multivariate polynomial regression equations to predict the value of the physical properties of crude oil from existing data then classify it into three categories using Naive Bayes, i.e., light oil, medium oil, and heavy oil. The physical properties of petroleum measured in the research are viscosity, interfacial tension, and density. The model uses laboratory data which are taken from the test results of Pertamina's KW-55 well as validation. The validation result shows that Multivariate Polynomial Regression has succeeded in predicting the value of viscosity, interfacial tension, and density with error values ranging from 0% to 1% from the sample data. With a low error value, the application can make forecasting with more variable conditions. The model still cannot be used independently without the Python environment, so to be used easily by more users, the model must be built into an independent application that can be installed on the user's device. In this research, the prediction application of petroleum physical properties has been built. The application is made using the Multivariate Polynomial Regression method according to the model in the previous study to predict the physical properties of petroleum, then uses Naive Bayes to classify the oil. The application completed the several adjustment to shift from model to application, including user interface, system, and database adjustments.

Keywords: EOR; petroleum physical properties; prediction

I. INTRODUCTION

In previous studies, researchers at the EOR laboratory of UPN Veteran Yogyakarta have developed biosurfactants to support the EOR process on oil samples taken from the KW-58 well of Pertamina (an oil company owned by the Indonesian government) (Sulistyarso, 2019).

In the process, every time a biosurfactant is added and the temperature is changed, measuring the physical properties of crude oil will be repeated. The prediction model for the petroleum physical properties was developed with data from EOR experiments that have been carried out to assist EOR researchers in the laboratory to obtain the number of physical properties of petroleum in accordance with the input data to reduce repeated experiments.

A machine learning-based model that can measure the physical properties of petroleum prediction from oil sample datasets from KW-55 Pertamina's well has been successfully built in the development stage after the first study. In this model, two methods are used, i.e., Multivariate Polynomial Regression and Naive Bayes Classifier (Sulistyarso, 2020). Multivariate Polynomial Regression is used because the data being modeled is multidimensional and non-linear, such as calculating viscosity predictions. In that process involves temperature data and biosurfactant data, as well as in calculating IFT and density. At the same time, the Naive Bayes Classifier is used to make predictions in classifying oil data into three categories, i.e., light oil, medium oil, and heavy oil. The model was done in Python programming, and it is a prototype that can only be used on the computer used to build the model. At the same time, this application needs to be distributed to the user's widely to install the application and use it on his computer independently. A system, product, or component's portability is defined as the degree of efficacy and efficiency with which it may be transferred from one hardware, software, or other operational or consumption environment to another. The essential purpose of software portability is to enable the action of porting an application from its current environment to a new or target environment before anything else (Mooney, 2004).

This study develops the model to an independent desktop-based application that could be distributed to EOR researchers and installed independently on each computer. The shift from a model to an application requires many adjustments, including the user interface design, database design, and system flow. Standard languages, system interface standards, portable libraries and compilers, and other well-known ways for ensuring portability are only a few examples. These tools are useful, but they aren't a replacement for a consistent portability strategy during development. The challenges are exacerbated by the more demanding needs of many modern software, such as timing (Mooney, 2004). When porting apps, time and cost are two crucial factors to consider. When porting apps, time and cost are two crucial factors to consider. Whether or not the software developed will save time and money (Obermeyer, 1996).

II. METHODOLOGY

There are a variety of approaches and techniques for making a model or software unit more portable. Three key ideas will effectively lead these techniques (Mooney, 2004), i.e.:

A. Control the interfaces

By creating a consistent set of interfaces between a software entity and all aspects of its environment, the major difficulties of portability can be overcome. This collection could comprise a wide range of interfaces of various types and levels. Because of differences in the components, the commonality of interfaces necessitates that each interface is designed to look the same from the perspective of the software being built. Many alternative tactics can be used to reach this goal. Some of these approaches may succeed in establishing a standard interface during development, while others will necessitate additional effort during porting. Porting software from model to desktop-based application in this study refers to the model's user interface. Still, it has to meet user requirements regarding the application's functions and ease of installation and use.

B. Isolate Dependencies

There will be aspects in a realistic software project that are dependent on their surroundings because the variations are too vast or crucial to be disguised by a single common interface. Because this is the component of the software that may require modification during porting, these elements must be limited to a minimal portion of the code.

For example, the software may has dependency on the memory or architecture of the operating system; graphics algorithms may be affected by the output models offered; high-performance parallel algorithms may be affected by the machine's architectural class.

C. Think Portable

Keep the portability goal during all the designs and development stages.

Before put the strategies work, the most important is how to address the portability issue during the software development process. Refers to the software development life cycle, such as the waterfall model (Sommerville, 2000), each significant activity in the lifecycle is tied to a set of specific portability challenges:

1. Specification
   Portability issues related to specifications are regarding the fulfillment of functional requirements and non-functional requirements during porting activities.
2. Design
   The heart of software development is design. As reflected in the specification, the construction of software architecture has to match with what the software is supposed to perform. A software design should, in theory, be independent of any implementation and hence entirely portable. In actuality, the design choice will have a significant impact on portability.
3. Implementation
   The process of turning a design into a functional software product is known as implementation. Even if the design is not expressly portable, it should not be platform-specific if good design practice has been

followed. The implementation of the software porting in this study will be in two kinds of platforms, i.e., windows-based and mac-os based. The portability issue in the implementation stage is the dependencies of the two platforms are quite different.

III. RESULTS AND DISCUSSION

According to the software development life cycle in the methods section, in this section, we will discuss the porting process for predicting the physical properties of petroleum from the previously built model.

3.1 Specification
Functional and non-functional requirements are included in specifications. Functional requirements are features that the application must meet, such as:
a. Can predict the viscosity data
b. Can predict the IFT data
c. Can predict the density data
d. Can save prediction results data
e. Can see the distribution of IFT training data
f. Can see the distribution of the viscosity training data
g. Can see the distribution of density training data
h. Can see the distribution of IFT test data
i. Can see the distribution of viscosity test data
j. Can see the distribution of density test data

Non functional requirement is limitations of the system's services or functions, such as time restrictions, development process constraints, and standardization. Non-functional requirements in this software are:
a. For Windows-based application
   System requirements:
   Minimum RAM 2GB
   Minimum Screen Resolution 1000*720
   Processor Minimum Core i3
   Does not require internet connection
   Minimum provides about 200 MB of storage.
   Windows 10

b. For MacOS-based application
   System requirements:
   Minimum RAM 2GB
   Minimum Screen Resolution 1000*720
   Processor Minimum Core i3
   Does not require internet connection
   Minimum provides about 100 MB of storage.
   Mac Mojave

3.2 Design
This process focuses on data structure, software architecture, interface representation.
a. Database Design
   The following is a representation of the database design used in the application, where the plan includes:
- Tbl_category functions as a container for the category of petroleum properties according to the value of the existing earth properties.

- Tbl_model represents the data model used, such as IFT, Density, or viscosity in the application.
- Tbl_model_data represents the data model used, namely Training, Testing, and Validation data on the application, functions as the data category.
- Tbl_data represents the results of the predicted data and also the training data in the application.

The relationship in database design below includes:

• For each tbl_category contains one or more data in each tbl_data

• For each tbl_model has one or more data in each tbl_data

• For each tbl_model has one or more data in each tbl_data

• As for multiple data, it must have one tbl_model_data, tbl_model, tbl_categories.

Figure 1 represents the entity relationship diagram for database design.
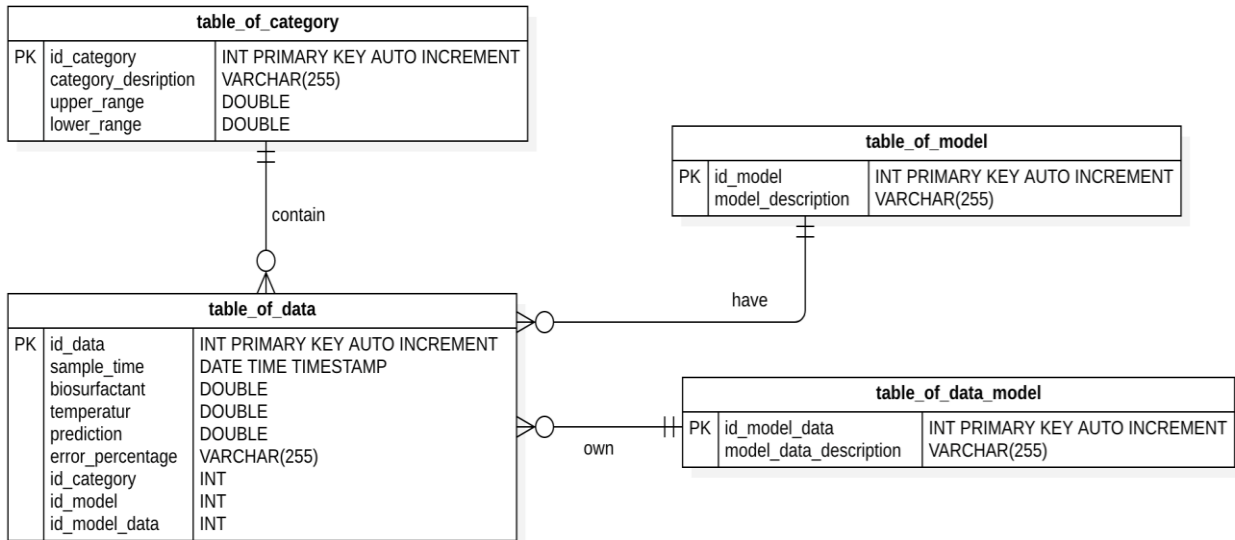
Figure. 1 Entity Relationship Diagram

b. System Design
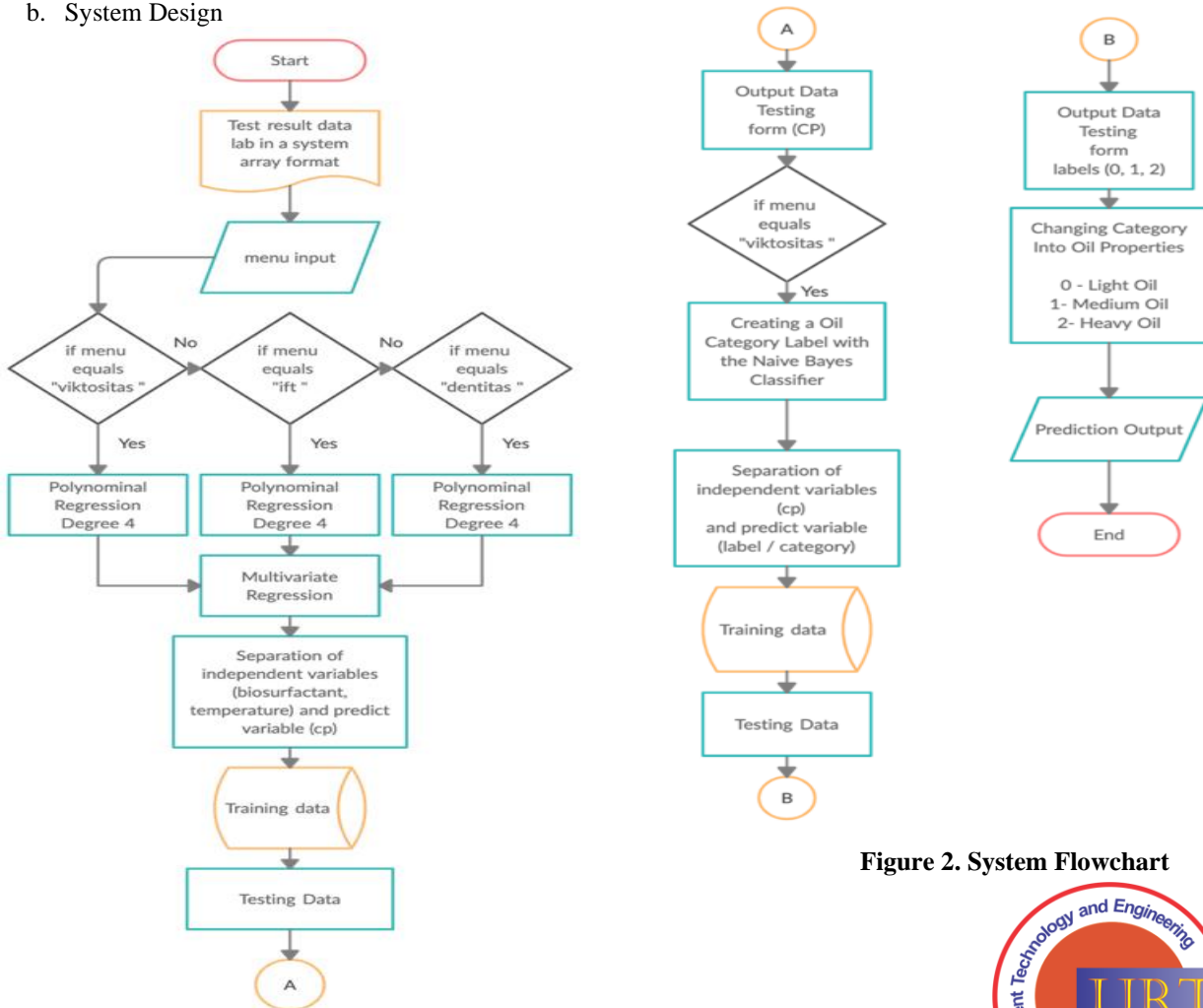
Figure 2. System Flowchart

141

A flowchart describes the design of the systemflow in figure 2. The system will start to test the raw data from the laboratory in array format, then user can choose one option of the menus. After the choice is made, the system will perform calculations using multivariate polynomial regression to get prediction results from the entered data. The prediction result will be classified by the Naïve Bayes in the system into three types of oil, i.e. light oil, medium oil and heavy oil.

c. User Interface Design

In this section, several mock-up designs for the prediction of petroleum physical properties application are presented.
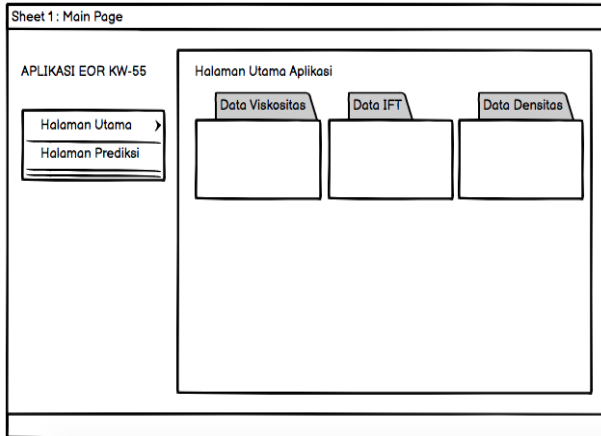
Figure 3. Main Menu Mock-up
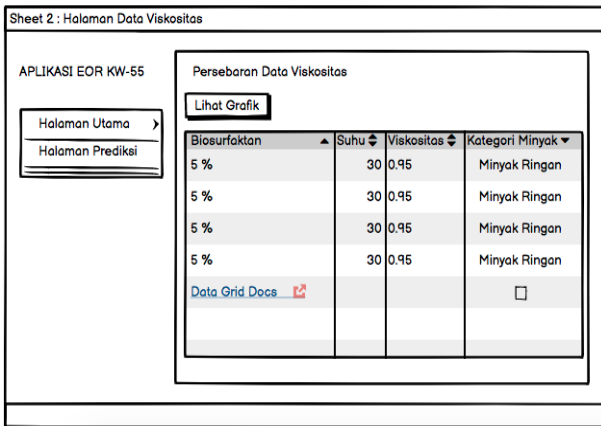
Figure 4. Viscosity Data Distribution Mock-up
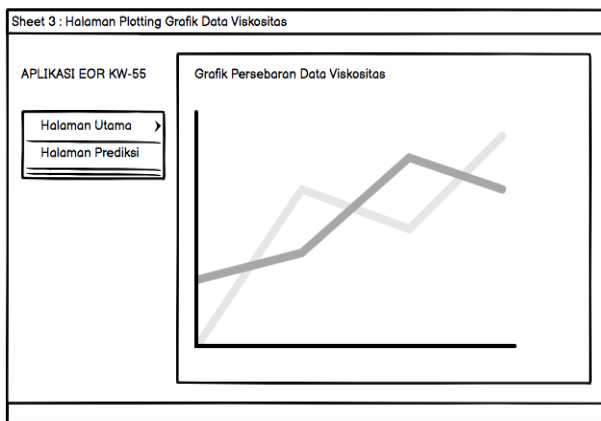
Figure 5. Viscosity Data Distribution Graphic Mock-up
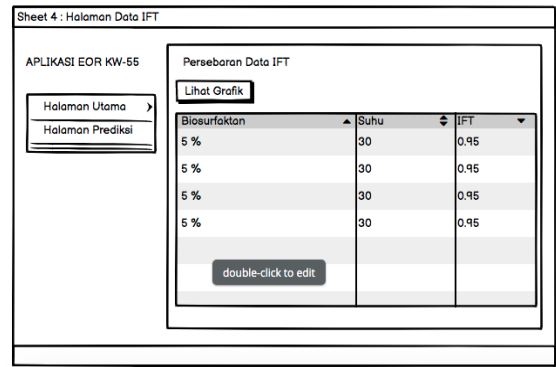
Figure 6. IFT Data Distribution Mock-up
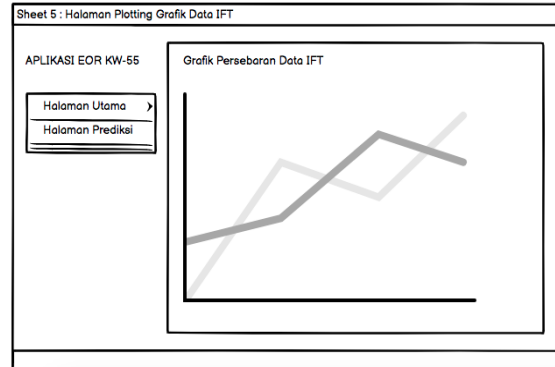
Figure 7. IFT Data Distribution Graphic Mock-up
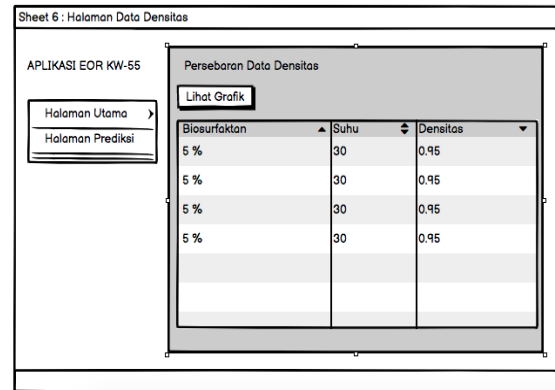
Figure 8. Density Dat Distribution Mock-up
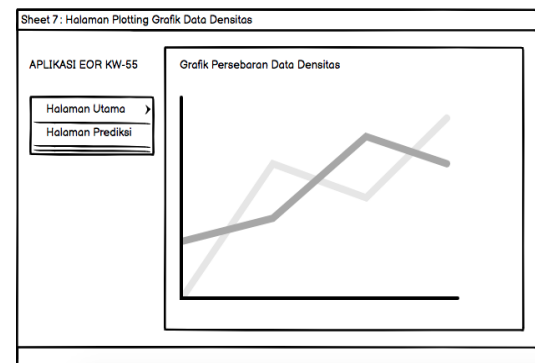
Figure 9. Density Data Distribution Graphic Mock-up

142
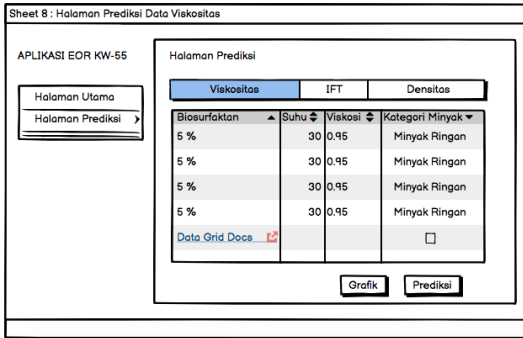
Figure 10. Viscosity Prediction Mock-up
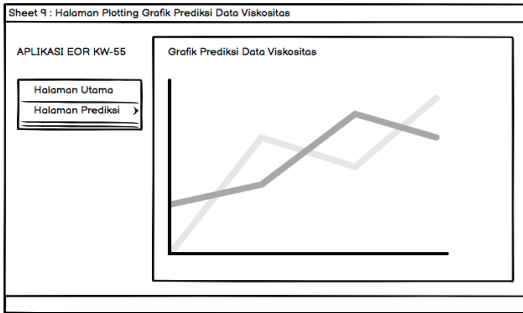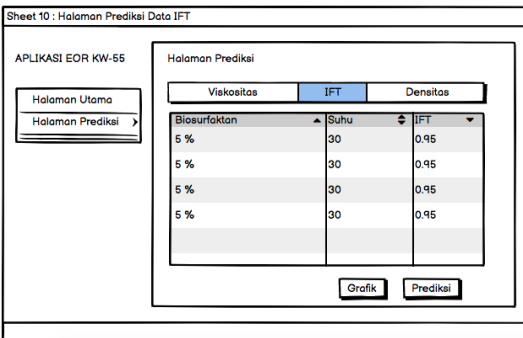

Figure 11. Viscosity Prediction Result Graphic Mock-up
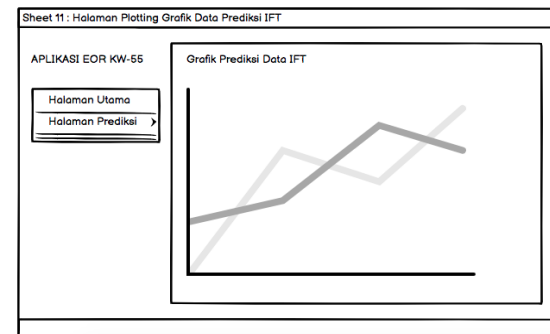

Figure 12. IFT Prediction Mock-up

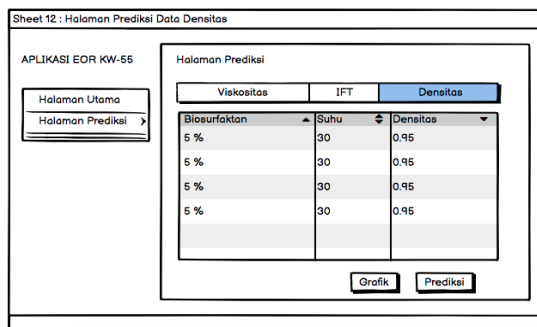**Figure 13. IFT Prediction Result Graphic Mock-up**

Figure 14. Density Prediction Mock-up

3.3 Implementation

Portability issues related to implementation are dependencies. The examples of dependency issue that occurs when porting an application are as follows :

a. Install dependencies on the system that will build the application using .dmg, namely on MAC or Unix operating systems. The following are the dependencies required by the application that is made, among others:

143

The encoding-tools function is used for encoding the built application. The function of matplotlib is to represent the data used in graphical form. The function of numpy is used for mathematical data operations with array storage. The function of pandas is to process data in the form of data sets according to the data modeling used. The pyinstaller function is used to build applications in the form of .dmg or .exe. PyQt5 functions are used for application dependencies as desktop implementations. The scikit-learn function is used for data prediction, and training data modeling in applications.

b. Build the application in the .spec model as a first step in building the application to *.exe or *.dmg format that is by using code :

```
pyi-makespec--onefile main_apps.py
```

The following is a form of a .spec file that has been built automatically :

```
# -*- mode: python ; coding: utf-8 -*-
block_cipher = None

added_files = [
    ("assets","assets")
]

A = Analysis(['main_apps.py'],
             pathex=['/Users/jazulienux/D
    esktop/Research Application'],
             binaries=[],
             datas=added_files,
             hiddenimports=[],
             hookspath=[],
             hooksconfig={},
             runtime_hooks=[],
             excludes=[],
             win_no_prefer_redirects=Fal
             se,
             win_private_assemblies=Fals
             e,
             cipher=block_cipher,
             noarchive=False)
pyz = PYZ(a.pure, a.zipped_data,
             cipher=block_cipher)

exe = EXE(pyz,
          a.scripts,
          a.binaries,
          a.zipfiles,
          a.datas,
          [],
          name='main_apps',
          debug=False,
          bootloader_ignore_signals=Fal
          se,
          strip=False,
          upx=True,
          upx_exclude=[],
          runtime_tmpdir=None,
          console=False,
          disable_windowed_traceback=Fa
          lse,
          target_arch=None,
          codesign_identity=None,
          entitlements_file=None )
app = BUNDLE(exe,
```

```
             name='main_apps.app',
             icon=None,
             bundle_identifier=None)
```

c. Build applications in the .spec model into .exe or .dmg on the operating system used in the following way :

```
pyinstaller --onefile main_apps.spec
```

```
hdiutil create ./Application-EOR.dmg -
srcfolder main_apps.app -ov
```

The results of the application build in question are represented in figure 17.
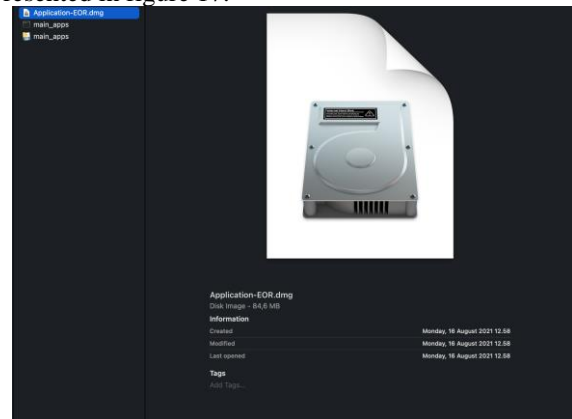

Figure 17. The Application

d. Run the application that was built whether it runs properly or not, and the figure below is a representation of running the application according to the build that has been done. Figure 19 shows the installed application from the built installer.
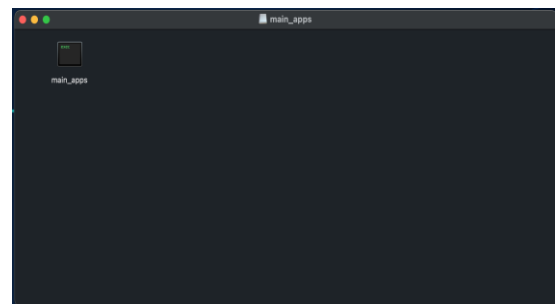

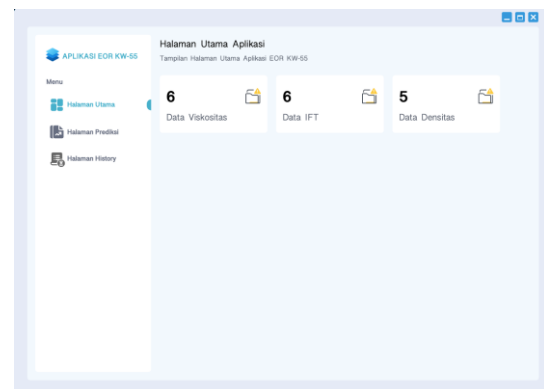**Figure 18. Running the Installation**

Figure 19. Installed Application

Figure 20 until 22 show the windows to display containing viscosity, IFT, and density training data. Each windows has button to show the graphic from the data. Figure 23 until 25 show the windows to display the graphic of Viscosity, IFT, and density training data.
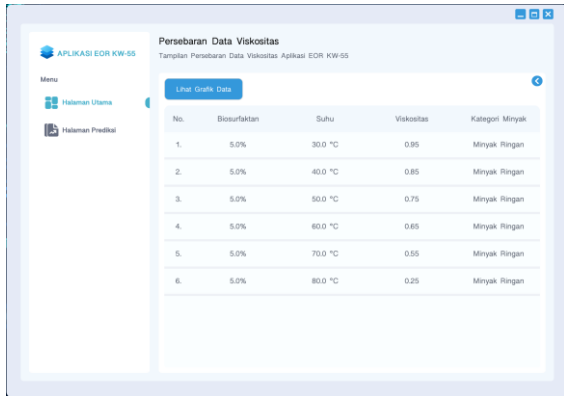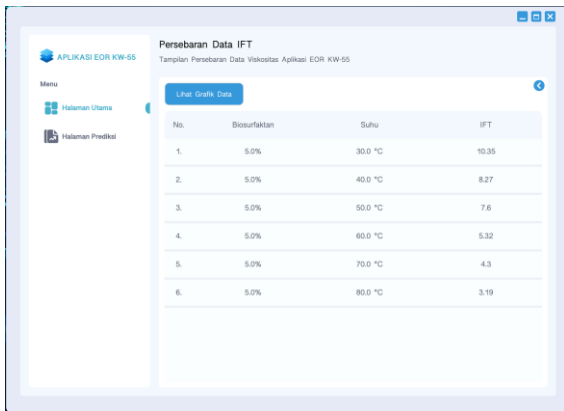

Figure 20. Viscosity Data Display
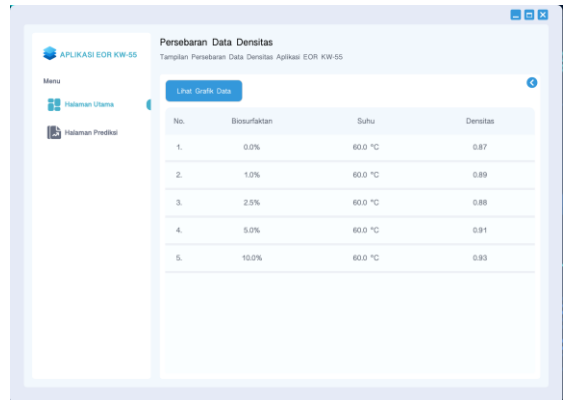

Figure 21. IFT Data Display

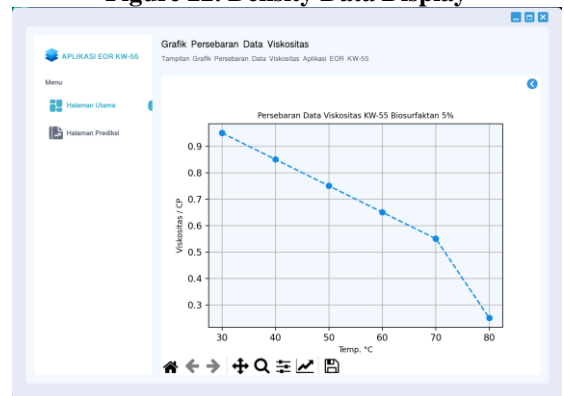**Figure 22. Density Data Display**

Figure 23. Viscosity Data Graphic Display
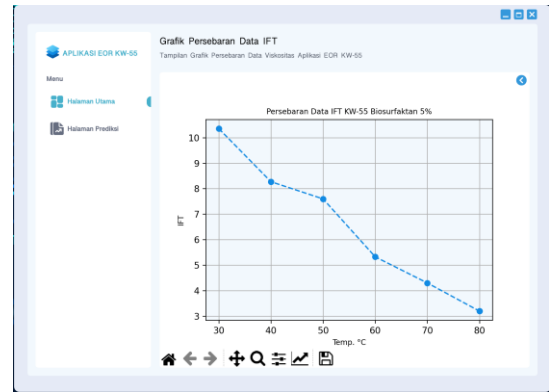

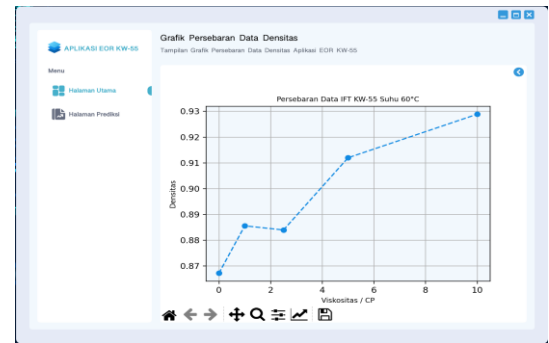**Figure 24. IFT Data Graphic Display**

Figure 25. Density Data Graphic Display

Prediction of the physical properties of petroleum in the built application is shown in figure 26 to figure 28 below. The graphic representation for prediction results is shown in figure 29 to figure 31.
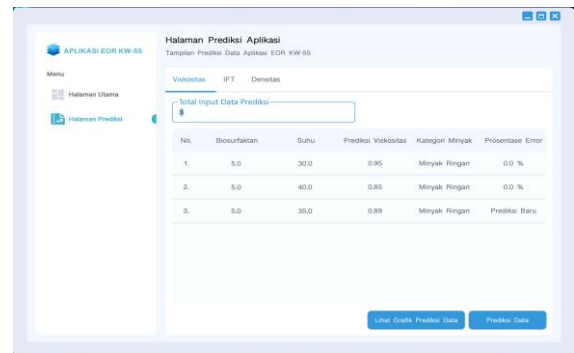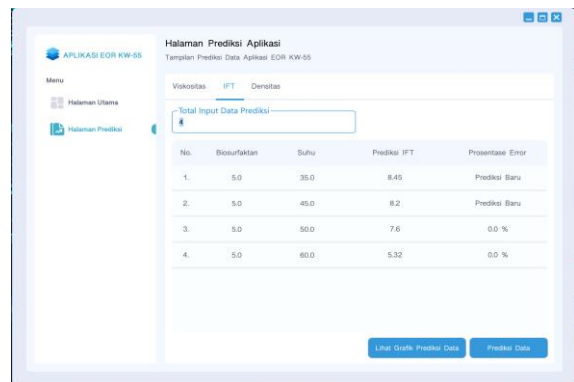

Figure 26. Viscosity Prediciton Result

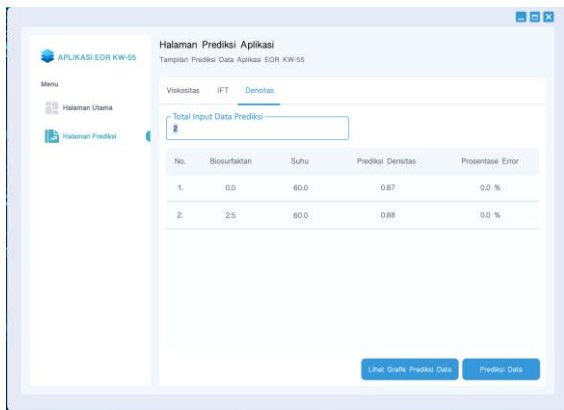
Figure 27. IFT Prediciton Result

Figure 28. Density Prediciton Result
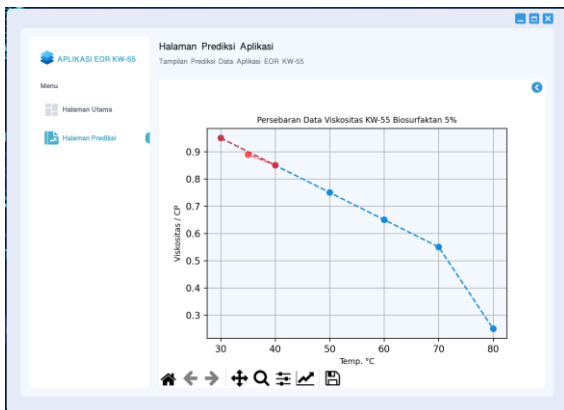

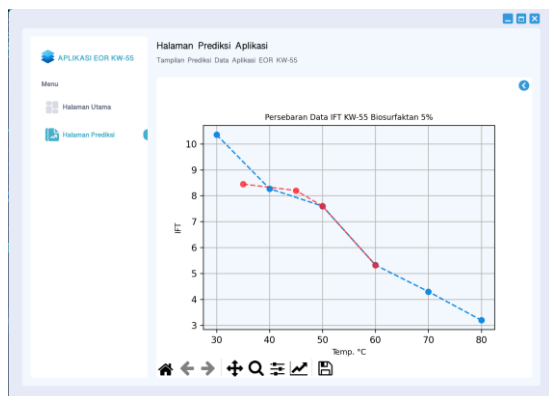**Figure 29. Viscosity Prediciton Result Graphic**

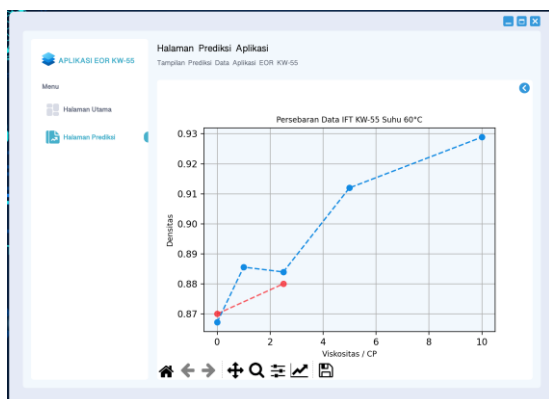**Figure 30. IFT Prediciton Result Graphic**

Figure 31. Density Prediciton Result Graphic

IV.   CONCLUSION

1. The validation result shows a low error value on Multivariate Polynomial Regression, so that the application can make forecasting with more variable conditions.

2. This application can be working easier and faster to get the result from density, viscosity, and IFT prediction.

3. The prediction application was made user-friendly so that can be installed and used on the computer independently.

ACKNOWLEDGEMENTS

REFERENCES

1. Bustami., (2013), Penerapan Algoritma Naive Bayes Untuk Mengklasifikasi Data Nasabah Asuransi, TECHSI : Jurnal Penelitian Teknik Informatika, Vol. 3, No.2, Hal. 127-146.
2. Irawati DA, Syaifudin YW, Tomasila FE, Setiawan A, Rohadi E. 2018. "Development of Android-based Rabbit Disease Expert System". International Journal of Engineering and Technology. 7(4.44):82-87.
3. Kementerian ESDM. 2015. Rencana Strategis 2015-2019. Direktorat Jenderal Minyak dan Gas Bumi Kementerian Energi dan Sumber Daya Mineral Indonesia.
4. MooneyJ.D.2004.DevelopingPortableSoftware.InInformationTechnology, Ricardo Reis (Ed.). IFIP International Federation for Information Processing, Vol. 157. Springer US, 55–84. http://dx.doi.org/10.1007/1-4020- 8159- 6_3
5. Robert T dan Haddad HH. 2005. Application Development for Desktop and PDA-style Device: Issues and Lessons Learned. Proceeding of The 43th ACM Southeast Conference, Vol 2:242-247. Pages 242–247. https://doi.org/10.1145/1167253.1167309.
6. Vural MS dan Gök M. 2017. Criminal Prediction using Naïve Bayes Theory. ACM Journal Neural Computing and Applications. 28(9).
7. Saleh, A. 2015. Implementasi Metode Klasifikasi Naïve Bayes Dalam Memprediksi Besarnya Penggunaan Listrik Rumah Tangga Citec Journal, Vol. 2, No. 3, ISSN: 2354-5771
8. Sarkar S dan Sriram Ram S. 2001. Bayesian Models for Early Warning Bank Failures. Management Science Journal. 47(11).
9. Satter A, Varnon JE, dan Hoang MT. 1994. Integrated Reservoir Management. Journal of Petroleum Technology. 46(12):1057-1064.
10. Sen, R. 2008. "Biotechnology in petroleum recovery:The Microbial EOR". Progress in Energy and Combustion Science Journal. 34(6):714-724.
11. Sommerville, I. 2000, Software Engineering, 6th ed, Addison-Wesley, Reading, Mass.
12. Sulistyarso HB, Irawati DA, Pamungkas J, Widiyaningsih I. 2020. Modeling of Crude Oil Types Classification Using the Naive Bayes Classifier Method. Proceeding of LPPM UPN "Veteran" Yogyakarta Conference Series 2020–Engineering and Science Series: 328-339. LPPM UPNVY. Yogyakarta.
13. Sulistyarso HB, Pamungkas J, Gusmarwani SR, Wahyuningsih T. 2019. Aplikasi Biosurfaktan dalam Upaya Peningkatan Perolehan Minyak Tahap Lanjut : Uji Laboratorium pada Sampel Sumur KW-58. Prosiding SNCP 2019 dengan Tema Pengembangan Ristek dan Pengabdian menuju Hilirisasi Industri: 55-60. LPPM UPNVY. Yogyakarta
14. Tisdall T, Chobharkar P, dan Kim D. 2018. Get Mobile: GetMobile: Mobile Computing and Communications. https://doi.org/10.1145/3229316.3229324.
15. Obermeyer L, Warshaw L, dan Miranker Daniel P. 1996. "Porting an Expert Database Application to An Active Database : An Experience Report". CIKM 96 : Proceeding of The Workshop on Databases: active and real time:37-40.

Retrieval Number: 100.1/ijrte.D65721110421
DOI: 10.35940/ijrte.D6572.1110421
Journal Website: www.ijrte.org

146

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)
© Copyright: All rights reserved.

AUTHORS PROFILE

Harry Budiharjo Sulistyarso. is Assosiate Professor in Petroleum Engineering Department UPN "Veteran",Yogyakarta. At the moment, he is active as lecturer and researcher in Enhanced Oil Recovery. Received his doctorate degree (S3) in Petroleum Engineering ITB in 2007. He obtained Bachelor degree (S1) in Petroleum Engineering from UPN "Veteran" Yogyakarta in 1990, and Master's degree (S2) in Petroleum Engineering from ITB in 1997. Professional Organization joined: The Institution of Engineers Indonesia (PII), Society of Indonesian Petroleum Engineers (IATMI), Indonesian Petroleum Association (IPA) and Society of Petroleum Engineers (SPE).

Dyah Ayu Irawati, ST., M.Cs., lahir di Klaten, 8 Juli 1984. Lulus dari SMAN 1 Klaten tahun 2002 lalu menyelesaikan pendidikan sarjana di Teknik Informatika UPN Veteran Yogyakarta pada tahun 2005. Setelah itu, menyelesaikan pendidikan magister di MIPA UGM tahun 2009. Pada tahun 2008 bekerja sebagai dosen di Jurusan Teknologi Informasi Politeknik Negeri Malang hingga tahun 2019, lalu pada Januari 2020 berpindah homebase ke Program Studi Sistem Informasi UPN "Veteran" Yogyakarta hingga sekarang.

Joko Pamungkas. Received his B.E. from UPN "Veteran" Yogyakarta. Received his M.Eng. from Bandung Institute of Technology (ITB) Mr. Joko Pamungkas become lecturer in Petroleum Engineering Department UPN "Veteran" Yogyakarta since 1993. Mr. Joko Pamungkas also become a consultant in petroleum industry for more than 25 years. Experienced in oil and gas plan of development (POD). Professional Organization joined: The Institution of Engineers Indonesia (PII), Society of Indonesian Petroleum Engineers (IATMI), Indonesian Geothermal Association (INAGA), and Indonesian Petroleum Association (IPA).

Indah Widyaningsih. Received her B.E. from UPN "Veteran" Yogyakarta. Received her M.Eng. from Bandung Institute of Technology. Worked at JOB Pertamina-Medco E&P Tomori for 4.5 years. Joined research at Ogrindo, ITB for 6 years and become a lecturer in Petroleum Engineering Department UPN "Veteran" Yogyakarta since 2013. Professional Organization joined: Society of Indonesian Petroleum Engineers (IATMI).