DATA ANALYTIC TOOL FOR EMERGENCY ROOMS

A PROJECT REPORT

Submitted by

NAVEEN KUMAR K (2116210701174)

NAVYA BALASUNDARAM (2116210701177)

OVIYA K (2116210701186)

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

RAJALAKSHMI ENGINEERING COLLEGE

ANNA UNIVERSITY, CHENNAI

MAY 2024

RAJALAKSHMI ENGINEERING COLLEGE, CHENNAI

BONAFIDE CERTIFICATE

Certified that this Thesis titled **"DATA ANALYTIC TOOL FOR EMERGENCY ROOMS**" is the bonafide work of "**NAVEEN KUMAR K (2116210701174), NAVYA BALASUNDARAM (2116210701177), OVIYA K (2116210701186)"** who carried out the work under my supervision. Certified further that to the best of my knowledge the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

SIGNATURE

Dr . K.Ananthajothi M.E.,Ph.D.,

PROJECT COORDINATOR

Professor

Department of Computer Science and Engineering

Rajalakshmi Engineering College

Chennai - 602 105

Submitted to Project Viva-Voce Examination held on_____

Internal Examiner **External Examine**r

ABSTRACT

With growing population, hospitals find it difficult to operate seamlessly, especially in emergency rooms due to proportional growing data. The number of patients as well as the need for resources, medicines and clinical supplies has increased greatly. Furthermore, with greater increase in data, arises a need for better analysis and visualization of such data. Only when the data is analysed as statistics, hospitals can make better decisions for a more efficient functioning of the facility.

Our goal is to give hospitals the tools they need to effectively analyze this data as statistics, enabling them to decide how best to allocate their resources and improve overall operational effectiveness. Our web application takes input from the hospital emergency room staff of various patient data such as gender, age, triaging, vitals during admission, comorbidities, geographic location of their residence, diagnosis, referral department, whether there was an occurrence of an accident/natural disaster and other valuable information. Our tool helps with analyzing and visualizing these on various scales and criteria against one another. It will give numerous key take-aways to the hospital on what to expect from any new incoming patient and how to care for them more effectively. If there is an unexpected visible pattern or trend seen in the statistics, then our tool will alert the healthcare management.

ACKNOWLEDGMENT

First, we thank the almighty god for the successful completion of the project. Our sincere thanks to our chairman **Mr. S. Meganathan B.E., F.I.E.,** for his sincere endeavor in educating us in his premier institution. We would like to express our deep gratitude to our beloved Chairperson **Dr. Thangam Meganathan Ph.D.,** for her enthusiastic motivation which inspired us a lot in completing this project and Vice Chairman **Mr. Abhay Shankar Meganathan B.E., M.S.,** for providing us with the requisite infrastructure.

We also express our sincere gratitude to our college Principal, **Dr. S. N. Murugesan M.E., PhD.,** and **Dr. P. KUMAR M.E., PhD, Director computing and information science , and Head Of Department of Computer Science and Engineering** and our project coordinator **Dr. K.Ananthajothi M.E.,Ph.D.,** for her encouragement and guiding us throughout the project towards successful completion of this project and to our parents, friends, all faculty members and supporting staffs for their direct and indirect involvement in successful completion of the project for their encouragement and support.

NAVEEN KUMAR K

NAVYA BALASUNDARAM

OVIYA K

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

This project offers a web application for a patient data analyzer. Our application uses Python and Dash for front end and backend backend, and will be working effectively. Users provide our application with a rich set of data that allow good visualization and provide with numerous statistics, based on their input of gender, age, triaging, vitals during admission, comorbidities, geographic location of their residence, referral department, whether there was an occurrence of an accident/natural disaster and other valuable information.

Our application processes the data using python and visualization using the advance techniques of Dash. Our application not only takes patient data input but also the resources used for them when they get admitted. Using this input, the application leverages Dash which is a power tool for creating visualizations of big data in a web application.

Our application facilitates the analysis and visualization of these against one other on different dimensions and criteria. The hospital will gain a great deal of insight from it about how to better care for newly admitted patients and what to anticipate from them. Our objective is to equip hospitals with the necessary tools to assess this data as statistics, so they may make the most informed decisions about resource allocation and enhance overall operational effectiveness. In addition to improving the management of the hospital's emergency room resources, this will aid in the process of making decisions about upcoming investments.

1.1 PROBLEM STATEMENT

With growing population, hospitals find it difficult to operate seamlessly, especially in emergency rooms due to proportional growing data. The number of patients as well as the need for resources, medicine etc has increased greatly. With greater increase in data, arises a need for better analysis and visualization of such data. Only when the data is analysed as statistics, hospitals can make better decisions for a more efficient functioning of the facility.

1.2 SCOPE OF THE WORK

Our project involves collecting basic patient information such as name, age, gender, and a few health-related indicators like hypertension and diabetes status. We also capture details about patient triage, location, and any recent accidents they may have had. This data is used to create simple visualizations and analyses to support basic decision-making within a healthcare environment, focusing on optimizing patient care and resource allocation. The goal is to provide actionable insights in a straightforward and efficient manner using the collected data.

1.3 AIM AND OBJECTIVES OF THE PROJECT

The aim of this project is to create and execute a system enabling hospitals to analyze and visualize data, tackling the difficulties brought about by an expanding population and a corresponding rise in the quantity of patients. The intention is to give hospitals the tools they need to analyze this data as statistics, enabling them to decide how best to allocate their resources and improve overall operational effectiveness. With Dash for visualization and for data administration, the system will give users immediate insights into important KPIs. It also provides a 24/7 real-time resource count for emergency rooms.

1.4 RESOURCES

This project has been developed through widespread secondary research of accredited manuscripts, standard papers, business journals, white papers, analysts' information, and conference reviews. Significant resources are required to achieve an efficacious completion of this project.

The following prospectus details a list of resources that will play a primary role in the successful execution of our project:

- A properly functioning workstation (PC, laptop, net-books etc.) to carry out desired research and collect relevant content.
- Unlimited internet access.
- Unrestricted access to the university lab in order to gather a variety of literature including academic resources.

1.5 MOTIVATION

Our data analytics application for hospital emergency rooms analyzes patient data and produces informative visuals using the Dash framework in Python. Our application generates understandable, interactive visual representations of complex information by processing incoming data from emergency room sources. Healthcare personnel can obtain useful insights into trends, patterns, and performance indicators inside their emergency departments by utilizing dynamic charts, graphs, and dashboards.

The program highlights important parameters like patient flow, resource use, and efficiency indicators, enabling hospital staff to monitor and improve emergency department operations. Decision-makers may swiftly pinpoint problem areas and make well-informed changes to improve patient care and expedite procedures with customizable visuals. Our technology helps hospitals to effectively adjust and optimize emergency services by utilizing data-driven insights, which eventually leads to better results.

CHAPTER 2

LITRETURE SURVEY

For our data analysis project using Dash for web-based visualization and exploration, a literature survey would encompass reviewing existing research, studies, and applications related to similar technologies and methodologies. This survey aims to gain insights into the current state-of-the-art practices, challenges, and innovations in the field of data visualization and web-based analytics.

In conducting the literature survey, we would start by exploring academic papers and research articles that discuss the use of Dash and related Python libraries like Plotly and Pandas for data analysis and visualization. This includes understanding the technical capabilities of these tools, their advantages, and any limitations or considerations highlighted in previous studies.

Additionally, we would delve into literature focusing on web-based data analytics, interactive dashboards, and user experience design in data visualization applications. By studying how other researchers and practitioners have approached similar projects, we can identify best practices, design patterns, and novel techniques that could inform our own project's methodology and implementation.

Furthermore, the literature survey would involve examining case studies and real-world applications where Dash and similar technologies have been successfully deployed. This helps us understand the practical implications of using these tools in different domains such as healthcare, finance, or logistics, and how they have contributed to data-driven decision-making and business intelligence.

In summary, the literature survey for our project serves as a foundational step to contextualize our work within the broader landscape of data visualization and web-based analytics. By synthesizing insights from existing literature, we can leverage proven methodologies and innovative approaches to develop a robust and impactful data analysis tool using Dash, contributing to the advancement of data science and analytics practices.

CHAPTER 3

SYSTEM DESIGN

3.1 GENERAL

In this section, we would like to show how the general outline of how all the components end up working when organized and arranged together. It is further represented in the form of a flow chart below.

3.2 SYSTEM ARCHITECTURE DIAGRAM

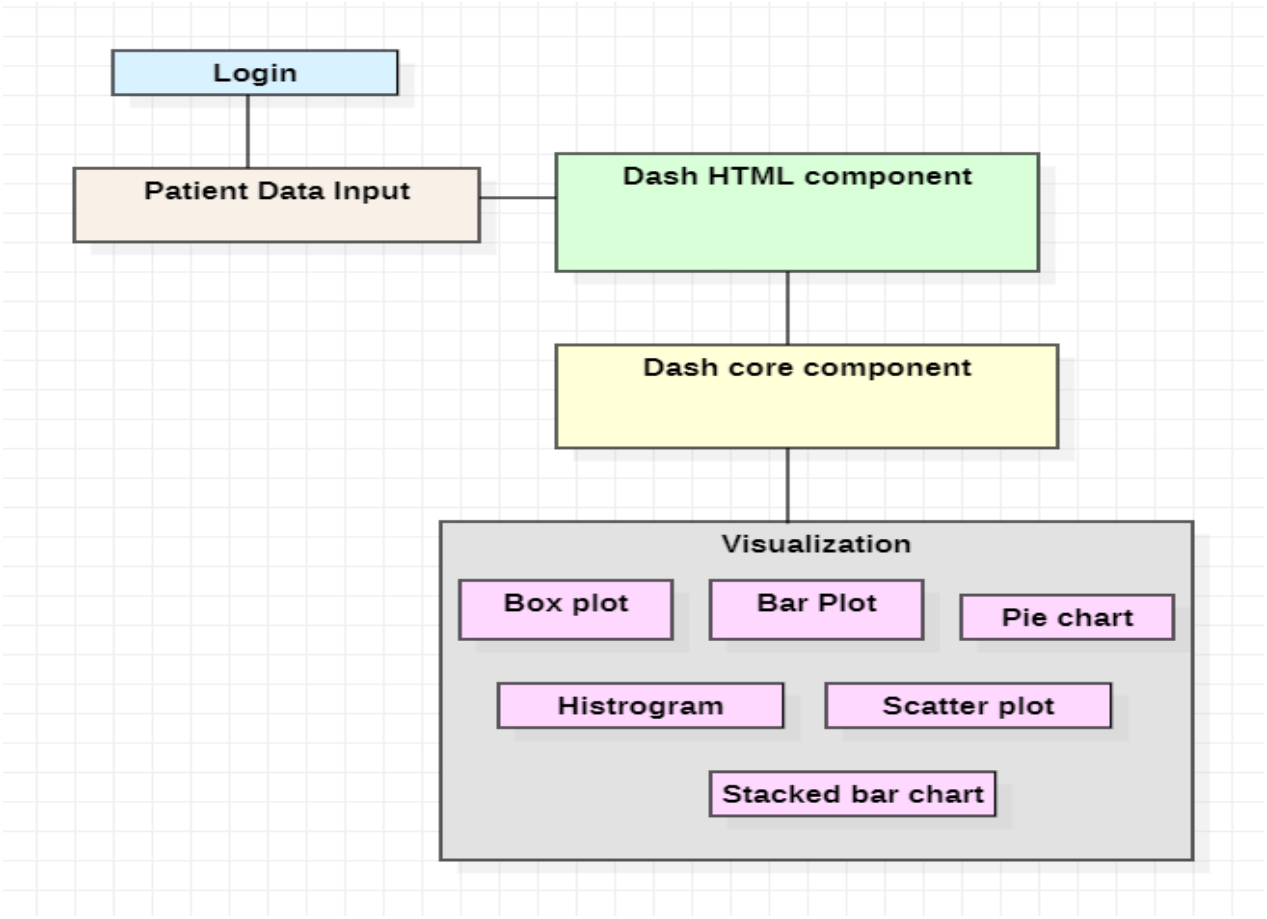

Fig 3.1: System Architecture

3.3 DEVELOPMENTAL ENVIRONMENT

3.31 HARDWARE REQUIREMENTS

The hardware requirements may serve as the basis for a contract for the system's implementation. It should therefore be a complete and consistent specification of the entire system. It is generally used by software engineers as the starting point for the system design.

Table 3.1 Hardware Requirements

COMPONENTS	SPECIFICATION
PROCESSOR	Intel Core i5
RAM	8 GB RAM
GPU	NVIDIA GeForce GTX 1650
MONITOR	15" COLOR
HARD DISK	512 GB
PROCESSOR SPEED	MINIMUM 1.1 GHz

3.31.1 SOFTWARE REQUIREMENTS

The software requirements document is the specifications of the system. It should include both a definition and a specification of requirements. It is aset of what the system should rather be doing than focus on how it should be done. The software requirements provide a basis for creating the software requirements specification. It is useful in estimating the cost, planning team activities, performing tasks, tracking the team, and tracking the team's progressthroughout the development activity.

Python IDLE or Visual Studio, Excel, and **Chrome** would all be required.

CHAPTER 4

PROJECT DESCRIPTION

4.1 METHODOLODGY

The methodology of our data analysis project using Dash involves leveraging Python libraries such as Pandas, Plotly, and Dash itself to create a dynamic web application for data visualization and analysis. Our approach begins with data collection from various sources, which is then cleaned and processed using Pandas to prepare it for analysis. This includes handling missing values, transforming data formats, and performing initial exploratory analysis to understand the dataset's characteristics.

Next, we utilize Plotly, a powerful graphing library, to generate interactive visualizations within the Dash framework. Plotly's capabilities allow us to create a wide range of charts and graphs, including scatter plots, bar charts, line plots, heatmaps, and more. These visualizations are integrated into Dash components such as graphs, tables, and dropdowns to build a user-friendly web interface.

The utility of Dash lies in its ability to democratize data analysis by providing a platform where stakeholders can interactively explore data insights. Through the web application, users can filter and drill down into specific datasets, toggle between different visualizations, and gain deeper insights into trends and patterns. This interactivity enhances decision-making processes by enabling real-time exploration of data, which is particularly useful for projects requiring rapid analysis and informed decision-making.

4.2 MODULE DESCRIPTION

4.2.1 Login Page

This page contains the form to fill the login credentials by the hospital authorities, since only that particular hospital will use it, there are no multiple users. An option to change the password is also provided to enhance security.

4.2.2 Patient Data Input Page

This page contains the form where the patient data is collected, it contains appropriate text fields as well as dropdown fields for easy filling. It collects important information such as age, presence of comorbidities, cause of accident if so, department they were referred to etc. This is the crucial data on which analysis will be done. All the fields are required to be filled without fail before submitting as everything is a crucial information.

4.2.3 Data Analytics Page

This is the main page where multiple graphs will be visible showing the analysis real time. It will be based on the patient data collected currently along with everything collected till date. It will show various graphs such as bar graphs, pie charts, scatter plots, histograms, stacked bar charts, and many other bivariate and multivariate graphs. These will be plotted on various different combinations of fields on all the axes. This will help the facility understand the current trend and make better decisions for the patients as well as the organization.

CHAPTER 5

RESULTS AND DISCUSSIONS

5.1 OUTPUT

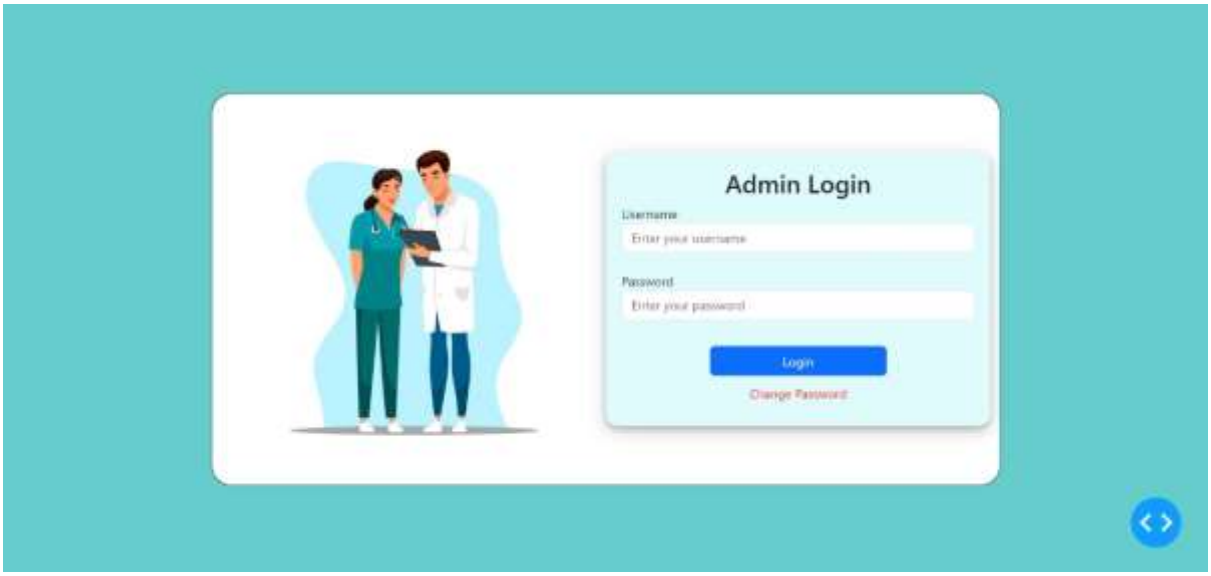

Fig 5.1: Admin Login

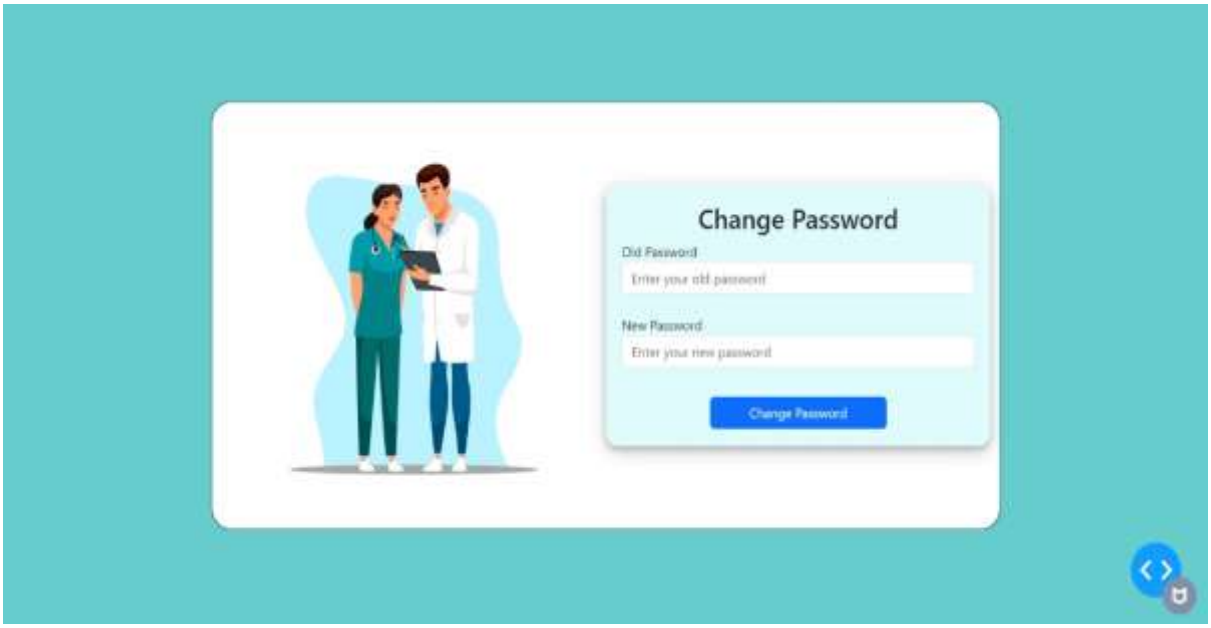

Fig 5.2:Change Password

Fig 5.3: Univariate Analysis

Fig 5.4: Bivariate Analysis

Fig 5.5:Multivariate Analysis

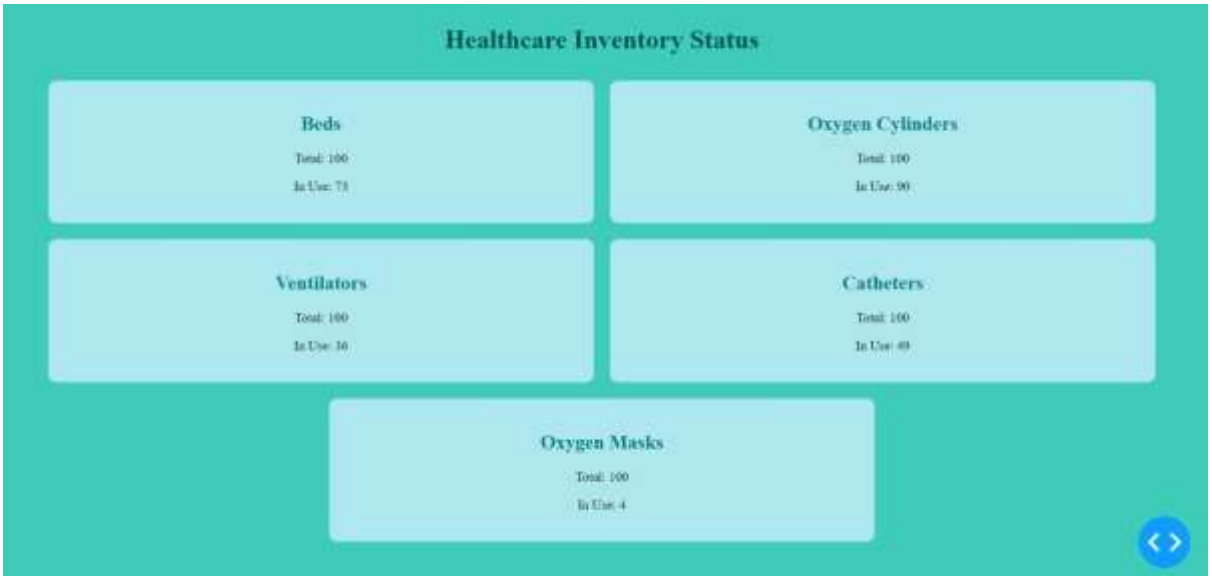

Fig 5.6: Inventory Status

5.2 RESULT

The outcome of this data analysis project will be a comprehensive web-based data visualization page featuring a diverse array of charts including bar graphs, pie charts, histograms, scatter plots, and various other univariate, bivariate, and multivariate visualizations. These interactive charts will allow users to explore different dimensions of the dataset, gaining insights into patterns, distributions, correlations, and relationships within the data. By leveraging Python libraries such as Plotly and Dash, the visualization page will offer a dynamic and user-friendly interface where stakeholders can interact with the data, apply filters, and drill down into specific details to make informed decisions.

Additionally, the project will deliver a dashboard that presents a summary of total resources alongside their utilization status. This dashboard will display key metrics such as the total count of resources available, the current number of resources in use or allocated, and utilization rates across different resource categories. This real-time view of resource utilization will enable stakeholders to monitor resource allocation efficiently, identify areas of optimization, and make strategic decisions based on actionable insights derived from the data analysis. Together, these deliverables will empower users with valuable tools for data exploration, visualization, and resource management within a dynamic web application framework.

CHAPTER 6

CONCLUSION AND FUTURE ENHANCEMENT

6.1 CONCLUSION

In conclusion, this data analysis project has successfully demonstrated the power and versatility of using Python-based technologies like Plotly and Dash to create effective data visualization tools and resource management dashboards. By developing a web-based visualization page with a rich assortment of charts including bar graphs, pie charts, histograms, scatter plots, and more, we have provided users with intuitive means to explore and understand complex datasets. These interactive visualizations enable stakeholders to uncover insights, identify trends, and make data-driven decisions.

Moreover, the implementation of a dashboard showcasing total resources and their utilization status adds a practical dimension to the project, offering a real-time snapshot of resource allocation and usage. This dashboard serves as a valuable tool for monitoring resource availability, optimizing utilization rates, and ensuring efficient resource management practices.

Overall, the project outcomes contribute to enhancing decision-making processes and operational efficiency within various domains by leveraging data visualization and dashboarding capabilities. Moving forward, this project lays a solid foundation for future endeavors in data analytics, highlighting the importance of interactive visualization tools and resource monitoring solutions in supporting informed decision-making and improving organizational outcomes.

FUTURE ENHANCEMENT

In considering future enhancements for our data analysis and visualization project, several key areas can be targeted to elevate the functionality and usability of the application. One significant enhancement involves expanding the repertoire of visualization tools by incorporating additional chart types. This could include integrating advanced charting techniques such as box plots, violin plots, radar charts, and network graphs. These sophisticated chart types offer deeper insights into complex data relationships and distributions, providing users with more comprehensive analytical capabilities.

Another crucial aspect of future enhancement involves enhancing interactivity within the visualizations. By implementing more advanced interactive features, users can dynamically interact with multiple charts simultaneously. Linked highlighting and brushing functionalities across different visualizations can facilitate seamless exploration and understanding of data correlations, empowering users to uncover meaningful insights more efficiently.

APPENDIX

SOURCE CODE:

```
import dash
from dash import dcc, html, Input, Output
import plotly.express as px
import pandas as pd
import plotly.graph_objs as go
import dash_bootstrap_components as dbc
from sqlalchemy import create_engine
import urllib.parse

def query_data():
    # Create a connection to your MySQL database
    password_encoded = urllib.parse.quote_plus("oviya@1604")
    engine = create_engine(f'mysql+mysqlconnector://root:{password_encoded}@localhost:3306/healthcare')

    # Query data from the database
    query = "SELECT * FROM patient_data"
    df = pd.read_sql(query, engine)

    # Close the database connection
    engine.dispose()
    return df


age_grp = {
    '1-17': (1, 17),
    '18-25': (18, 25),
    '26-35': (26, 35),
    '36-50': (36, 50),
    '51-70': (51, 70),
    '71-100': (71, 100),
}

def assign_age_group(age):
    for grp, (start, end) in age_grp.items():
```

```
        if start <= age <= end:
            return grp
    return 'Unknown'

app = dash.Dash(__name__, external_stylesheets=[dbc.themes.BOOTSTRAP])

app.layout = dbc.Container([
    dcc.Dropdown(
        id='dropdown',
        options=[
            {'label': 'Univariate', 'value': 'option1'},
            {'label': 'Bivariate', 'value': 'option2'},
            {'label': 'Multivariate', 'value': 'option3'}
        ],
        value='option1',
        style={"width": "50%"}
    ),
    html.Div(id='output-container')
], className="my-4")

@app.callback(
    Output('output-container', 'children'),
    [Input('dropdown', 'value')]
)
def update_output(selected_option):
    if selected_option == 'option1':
        df = query_data()
        df['Age_Group'] = df['Age'].apply(assign_age_group)
        accident_counts = df.groupby(['Gender', 'Accident']).size().reset_index(name='count')
        triage_colors = {'Yellow': 'Yellow', 'Green': 'Green', 'Red': 'Red'}

        return html.Div([
                dbc.Row([
                dbc.Col(dcc.Graph(
                id='histogram',
                figure=px.histogram(df, x='Age_Group', title='Age Distribution',
color_discrete_sequence=px.colors.qualitative.Safe)
            ), width=6),
                dbc.Col(dcc.Graph(
                    id='triage-vs-dept',
```

```
        figure=px.histogram(df, x=df['Triage'], title="Triage",  color='Triage',
            color_discrete_map=triage_colors
              )
    ), width=6),
]),
dbc.Row([
    dbc.Col(dcc.Graph(
        id='location',
        figure=px.histogram(df, x=df['Location'], title="Location",
                        color_discrete_sequence=px.colors.qualitative.Set3)
    ), width=6),
    dbc.Col(dcc.Graph(
        id='dept',
        figure=px.pie(df, names=df['Department'], title="Department",
                    color_discrete_sequence=px.colors.qualitative.Plotly)
    ), width=6)
]),
dbc.Row([
    dbc.Col(dcc.Graph(
        id='accident',
        figure=px.histogram(df, x=df['Accident'], title="Accident",
                        color_discrete_sequence=px.colors.qualitative.Bold)
    ), width=4),
    dbc.Col(dcc.Graph(
        id='diabetes',
        figure=px.histogram(df, x=df['Diabetes'], title="Diabetes",
                        color_discrete_sequence=px.colors.qualitative.Safe)
    ), width=4),
    dbc.Col(dcc.Graph(
        id='hyper',
        figure=px.histogram(df, x=df['Hypertension'], title="Hypertension",
                        color_discrete_sequence=px.colors.qualitative.Set3)
    ), width=4)
]),
dbc.Row([
    dbc.Col(dcc.Graph(
        id='heart',
        figure=px.histogram(df, x=df['History of Heart Disease'], title="Heart Disease",
                        color_discrete_sequence=px.colors.qualitative.Pastel2)
    ), width=4),
```

```
            dbc.Col(dcc.Graph(
                id="gender",
                figure=px.histogram(df, x=df['Gender'], title="Gender",
                                    color_discrete_sequence=px.colors.qualitative.Prism)
            ), width=4),
            dbc.Col(dcc.Graph(
                id="cause-of-accident",
                figure=px.histogram(df, x=df['Cause of Accident'], title="Cause of Accident",
                                    color_discrete_sequence=px.colors.qualitative.G10)
            ), width=4)
        ]),
                dbc.Row([
                      dbc.Col(dcc.Graph(
                id='age',
                figure=px.pie(df, names=df['Age_Group'], title="Age",
                              color_discrete_sequence=px.colors.qualitative.Pastel)
            ), width=6),
        dbc.Col(dcc.Graph(
            id='pie-chart',
            figure=px.pie(df, names='Gender', title='Gender Distribution')
        ), width=6)
    ])
    ])

  elif selected_option == 'option2':
    df = query_data()
    triage_colors = {'Yellow': 'Yellow', 'Green': 'Green', 'Red': 'Red'}
    return html.Div([
        dbc.Row([
            dbc.Col(dcc.Graph(
                id='triage-vs-dept',
                figure=px.box(df, x=df['Triage'], y=df['Department'], title="Triage vs
Departments",  color='Triage',
                      color_discrete_map=triage_colors
                       )
            ), width=6),
            dbc.Col(dcc.Graph(
                id='count-of-accidents',
                figure=px.bar(accident_counts, x='Gender', y='count', color='Accident',
                              title='Count of Accidents by Gender', barmode='stack')
```

```
            ), width=6)
        ]),
        dbc.Row([
            dbc.Col(dcc.Graph(
                id='cause-department',
                figure=px.bar(df, x='Department', color='Cause of Accident', title='Cause of Accident vs Department')
            ), width=6),
            dbc.Col(dcc.Graph(
                id='age-group-department',
                figure=px.bar(df, x='Department', color='Age_Group', title='Age Group vs Department')
            ), width=6)
        ])
    ])
elif selected_option == 'option3':
    df = query_data()
    df_filtered = df[df['Cause of Accident'] != '-']
    count_filtered = len(df_filtered)
    return html.Div([
    dbc.Row([
        dbc.Col(dcc.Graph(
            id='scatter-1',
            figure=px.scatter(df, x='Age', y='Department', color='Gender', title='Age vs Department by Gender')
        ), width=12),
        dbc.Col(dcc.Graph(
            id='scatter-2',
            figure=px.scatter(df_filtered, x='Age', y='Department', color='Cause of Accident',
                              title=f'Age vs Department by Cause of Accident (Count: {count_filtered})')
        ), width=12)
    ])
])

if __name__ == '__main__':
    app.run_server(debug=True)
```

```
import dash
from dash import Input, Output, State, html ,dcc
import pandas as pd
from sqlalchemy import create_engine
import urllib.parse

# Initialize the Dash app
app = dash.Dash(__name__)

# Define the layout of the app
app.layout = html.Div(
    style={
        'font-family': 'Arial, sans-serif',
        'background-color': '#d4f0f0',
        'padding': '40px 20px',  # Added more padding for spacing
        'align-items': 'center'  # Center-align the content
    },
    children=[
        html.Div(
            className='container',
            style={
                'max-width': '1000px',  # Increased max-width
                'margin': '0 auto',

                'padding': '40px',
                'border-radius': '10px',
                'box-shadow': '0 2px 8px rgba(0, 0, 0, 0.1)',
                'display': 'flex',
                'flex-direction': 'column',  # Align items in a column
                'align-items': 'center'  # Center align items horizontally
            },
            children=[
                html.H1('Patient Information Form', style={'color': 'A4EDDD'}),
                html.Form(
                    id='patient-form',
                    className='form',
                    children=[
                        html.Div(
                            className='form-group',
```

```
                style={'display': 'flex', 'align-items': 'center', 'margin-bottom': '20px'},
                children=[
                    html.Label('Name:', style={'font-weight': 'bold', 'margin-right': '10px'}),
                    dcc.Input(type='text', id='name', name='name', placeholder='Enter your
name', style={'flex': 1})
                ]
            ),
            html.Div(
                className='form-group',
                style={'display': 'flex', 'align-items': 'center', 'margin-bottom': '20px'},
                children=[
                    html.Label('Age:', style={'font-weight': 'bold', 'margin-right': '10px'}),
                    dcc.Input(type='number', id='age', name='age', placeholder='Enter your
age', style={'flex': 1})
                ]
            ),
            html.Div(
                className='form-group',
                style={'display': 'flex', 'align-items': 'center', 'margin-bottom': '20px'},
                children=[
                    html.Label('Gender:', style={'font-weight': 'bold', 'margin-right': '10px'}),
                    dcc.Dropdown(
                        id='gender',
                        options=[
                            {'label': 'Male', 'value': 'Male'},
                            {'label': 'Female', 'value': 'Female'},
                            {'label': 'Other', 'value': 'Other'}
                        ],
                        placeholder='Select gender',
                        style={'flex': 1}
                    )
                ]
            ),
            html.Div(
                className='form-group',
                style={'display': 'flex', 'align-items': 'center', 'margin-bottom': '20px'},
                children=[
                    html.Label('Hypertension:', style={'font-weight': 'bold', 'margin-right':
'10px'}),
                    dcc.Dropdown(
```

```
                        id='hypertension',
                        options=[
                            {'label': 'Yes', 'value': 'Yes'},
                            {'label': 'No', 'value': 'No'}
                        ],
                        placeholder='Select hypertension status',
                        style={'flex': 1}
                    )
                ]
            ),
            html.Div(
                classNam e='form-group',
                style={'display': 'flex', 'align-items': 'center', 'margin-bottom': '20px'},
                children=[
                    html.Label('Diabetes:', style={'font-weight': 'bold', 'margin-right': '10px'}),
                    dcc.Dropdown(
                        id='diabetes',
                        options=[
                            {'label': 'Yes', 'value': 'Yes'},
                            {'label': 'No', 'value': 'No'}
                        ],
                        placeholder='Select diabetes status',
                        style={'flex': 1}
                    )
                ]
            ),
            html.Div(
                classNam e='form-group',
                style={'display': 'flex', 'align-items': 'center', 'margin-bottom': '20px'},
                children=[
                    html.Label('History of Heart Disease:', style={'font-weight': 'bold', 'margin-right': '10px'}),
                    dcc.Dropdown(
                        id='heart_disease',
                        options=[
                            {'label': 'Yes', 'value': 'Yes'},
                            {'label': 'No', 'value': 'No'}
                        ],
                        placeholder='Select heart disease status',
```

```
                        style={'flex': 1}
                    )
                ]
            ),
            html.Div(
                className='form-group',
                style={'display': 'flex', 'align-items': 'center', 'margin-bottom': '20px'},
                children=[
                    html.Label('Triage:', style={'font-weight': 'bold', 'margin-right': '10px'}),
                    dcc.Dropdown(
                        id='triage',
                        options=[
                            {'label': 'Red', 'value': 'Red'},
                            {'label': 'Yellow', 'value': 'Yellow'},
                            {'label': 'Green', 'value': 'Green'}
                        ],
                        placeholder='Select triage level',
                        style={'flex': 1}
                    )
                ]
            ),
            html.Div(
                className='form-group',
                style={'display': 'flex', 'align-items': 'center', 'margin-bottom': '20px'},
                children=[
                    html.Label('Location:', style={'font-weight': 'bold', 'margin-right': '10px'}),
                    dcc.Input(type='text', id='location', name='location', placeholder='Enter location', style={'flex': 1})
                ]
            ),
            html.Div(
                className='form-group',
                style={'display': 'flex', 'align-items': 'center', 'margin-bottom': '20px'},
                children=[
                    html.Label('Accident:', style={'font-weight': 'bold', 'margin-right': '10px'}),
                    dcc.Dropdown(
                        id='accident',
                        options=[
```

```
                        {'label': 'Yes', 'value': 'Yes'},
                        {'label': 'No', 'value': 'No'}
                    ],
                    placeholder='Select accident status',
                    style={'flex': 1}
                )
            ]
        ),
        html.Div(
            className='form-group',
            style={'display': 'flex', 'align-items': 'center', 'margin-bottom': '20px'},
            children=[
                html.Label('Cause of Accident:', style={'font-weight': 'bold', 'margin-right': '10px'}),
                dcc.Dropdown(
                    id='cause',
                    options=[
                        {'label': 'Fall', 'value': 'Fall'},
                        {'label': 'Fire', 'value': 'Fire'},
                        {'label': 'Drowning', 'value': 'Drowning'},
                        {'label': 'Earthquakes', 'value': 'Earthquakes'},
                        {'label': 'Chemicals/Poisoning', 'value': 'Chemicals'},
                        {'label': 'Vehicle Crash', 'value': 'Vehicle Crash'},
                        {'label': '-', 'value': '-'}
                    ],
                    placeholder='Select cause of accident',
                    style={'flex': 1}
                )
            ]
        ),
        html.Div(
            className='form-group',
            style={'display': 'flex', 'align-items': 'center', 'margin-bottom': '20px'},
            children=[
                html.Label('Department:', style={'font-weight': 'bold', 'margin-right': '10px'}),
                dcc.Dropdown(
                    id='dept',
                    options=[
                        {'label': 'Ophthalmology', 'value': 'Ophthalmology'},
```

```
                        {'label': 'Cardiology', 'value': 'Cardiology'},
                        {'label': 'Emergency Medicine', 'value': 'Emergency Medicine'},
                        {'label': 'Orthopedics', 'value': 'Orthopedics'},
                        {'label': 'Pulmonology', 'value': 'Pulmonology'},
                        {'label': 'Endocrinology', 'value': 'Endocrinology'},
                        {'label': 'General Surgery', 'value': 'General Surgery'},
                        {'label': 'Obstetrics and Gynaecology', 'value': 'Obstetrics and Gynaecology'},
                        {'label': 'Pediatrics', 'value': 'Pediatrics'}
                    ],
                    placeholder='Select the Department',
                    style={'flex': 1}
                )
            ]
        ),
        html.Button(
            'Submit',
            id='submit-button',
            type='submit',
            style={'margin-top': '20px', 'padding': '10px',
                'background-color': '#4CAF50', 'color': '#fff',
                'border': 'none', 'border-radius': '4px',
                'cursor': 'pointer', 'font-size': '16px'}  # Increased margin-top
        ),
        html.Div(id='output-message', style={'margin-top': '20px', 'font-weight': 'bold','margin':'auto'})  # Adjusted margin-top
    ]
  )
 ]
 )
)
]
)

if __name__ == '__main__':
    app.run_server(debug=True)


# Callback to handle form submission and append data to Excel file
@app.callback(
    Output('output-message', 'children'),
```

```
    [Input('submit-button', 'n_clicks')],
    [
        State('name', 'value'),
        State('age', 'value'),
        State('gender', 'value'),
        State('hypertension', 'value'),
        State('diabetes', 'value'),
        State('heart_disease', 'value'),
        State('triage', 'value'),
        State('location', 'value'),
        State('accident', 'value'),
        State('cause', 'value'),
        State('dept', 'value')
    ]
)

def submit_form(n_clicks, name, age, gender, hypertension, diabetes, heart_disease, triage, location, accident, cause, dept):
    if n_clicks:
        # Check if all required fields are filled
        if not all([name, age, gender, hypertension, diabetes, heart_disease, triage, location, accident, cause, dept]):
            return html.Div('Please fill in all required fields.', style={'color': 'red'})

        password_encoded = urllib.parse.quote_plus("oviya@1604")
        # Create a connection to your MySQL database
        engine = create_engine(f'mysql+mysqlconnector://root:{password_encoded}@localhost:3306/healthcare')

        try:
            # Create a DataFrame with the new row data
            new_row_data = {
                'Name': name,
                'Age': age,
                'Gender': gender,
                'Hypertension': hypertension,
                'Diabetes': diabetes,
                'History of Heart Disease': heart_disease,
                'Triage': triage,
```

```
            'Location': location,
            'Accident': accident,
            'Cause of Accident': cause,
            'Department': dept
        }

        # Write the DataFrame to the MySQL database
        pd.DataFrame([new_row_data]).to_sql('patient_data', con=engine, if_exists='append',
index=False)

        # Close the database connection
        engine.dispose()

        return html.Div('Form submitted successfully! Data added to the MySQL database.')
    except Exception as e:
        return html.Div(f'Error: {str(e)}')

  return ''


if __name__ == '__main__':
  app.run_server(debug=True)
```

REFERENCES

[1] T. Anderson and S. Dragićević, "Complex spatial networks: Theory and geospatial applications", *Geography Compass*, vol. 14, no. 9, Sep. 2020

[2] M. Bastian, S. Heymann and M. Jacomy, "Gephi: An open source software for exploring and manipulating networks", Proc. Int. AAAI Conf. Web Social Media, vol. 3, no. 1, pp. 361-362, Mar. 2009.

[3] S. Batt, O. R. Harmon and P. Tomolonis, "Learning tableau: A data visualization tool", J. Econ. Educ., vol. 51, no. 3, pp. 317-328, 2020

[4] M. Behrisch, L. Bieker, J. Erdmann and D. Krajzewicz, "SUMO—Simulation of urban mobility—An overview", *Proc. 3rd Int. Conf. Adv. Syst. Simulation*, pp. 55-60, 2011.

[5] G. Boeing, "OSMnx: A Python package to work with graph-theoretic OpenStreetMap street networks", *J. Open Source Softw.*, vol. 2, no. 12, pp. 215, Apr. 2017.

[6] G. Boeing, "OSMNX: New methods for acquiring constructing analyzing and visualizing complex street networks", *Comput. Environ.*