2) a. Dashboard creation using visualization tools for the healthcare code.

Code :

```
import dash
import dash_core_components as dcc
import dash_html_components as html
from dash.dependencies import Input, Output
import pandas as pd
import plotly.express as px

# Sample healthcare data (replace with your data source)
data = pd.read_csv('healthcare_data.csv')

# Create a Dash web application
app = dash.Dash(_name_)

# Define the layout of the dashboard
app.layout = html.Div([
    html.H1("Healthcare Data Dashboard"),

    # Dropdown for selecting a specific metric
    dcc.Dropdown(
        id='metric-dropdown',
        options=[
            {'label': 'Metric 1', 'value': 'metric1'},
            {'label': 'Metric 2', 'value': 'metric2'},
            # Add more options as needed
        ],
        value='metric1'  # Default selected metric
    ),

    # Graph to display the selected metric
    dcc.Graph(id='metric-graph'),
])

# Define callback to update the graph based on the selected metric
@app.callback(
    Output('metric-graph', 'figure'),
    [Input('metric-dropdown', 'value')]
)
def update_graph(selected_metric):
    # Filter the data based on the selected metric
    filtered_data = data[data['metric'] == selected_metric]

    # Create a Plotly figure for visualization
    fig = px.bar(filtered_data, x='x-axis-column', y='y-axis-column', title=f'{selected_metric} Visualization')

    return fig

if _name_ == '_main_':
    app.run_server(debug=True)
```

2) b. Dashboard creation using visualization tools for the finance code.

Code :

```
import dash
import dash_core_components as dcc
import dash_html_components as html
from dash.dependencies import Input, Output
import pandas as pd
import plotly.express as px

# Sample finance data (replace with your data source)
data = pd.read_csv('finance_data.csv')

# Create a Dash web application
app = dash.Dash(_name_)

# Define the layout of the dashboard
app.layout = html.Div([
    html.H1("Finance Data Dashboard"),

    # Dropdown for selecting a financial metric
    dcc.Dropdown(
        id='metric-dropdown',
        options=[
            {'label': 'Stock Price', 'value': 'stock_price'},
            {'label': 'Market Cap', 'value': 'market_cap'},
            # Add more options as needed
        ],
        value='stock_price'  # Default selected metric
    ),

    # Graph to display the selected metric
    dcc.Graph(id='metric-graph'),
])

# Define callback to update the graph based on the selected metric
@app.callback(
    Output('metric-graph', 'figure'),
    [Input('metric-dropdown', 'value')]
)
def update_graph(selected_metric):
    # Filter the data based on the selected metric
    filtered_data = data[data['metric'] == selected_metric]

    # Create a Plotly figure for visualization
    fig = px.line(filtered_data, x='date', y='value', title=f'{selected_metric} Visualization')

    return fig

if _name_ == '_main_':
    app.run_server(debug=True)
```