In [ ]:

```
# SHARKS range of innovative, new generation fire fighting products and systems are des
# Smith & Sharks Projects (India) Pvt. Ltd., a company promoted by a team of India's le
# 40 years of outstanding performance in the field of Fire Protection Engineering and re

# This is my company's tender related price data, which we charge the client for instal
# And I want to apply some algorithm to this data, so that I can find out, how long wil
# total workers and materials may be required.
# So I can say that this my project, it can give information about the time and men powe
# prediction so that we can reduce the delay in the project.
```

importing the model which is required for this projects.

In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

read the data from excel sheet after converting csv format

In [2]:

```
sspl = pd.read_csv(r"C:\Users\91898\Desktop\7AM_Task/south-info-boq-electrical.csv")
df = pd.DataFrame(sspl)
df.describe()
```

Out[2]:

	QUANTITY	supply amount	Total Supply Amount\n(Including Taxes)	Errection Amount	Total
count	55.000000	55.000000	55.000000	55.000000	55.000000
mean	18.660000	29456.872727	33404.300594	3390.145455	38759.563636
std	52.568708	81411.015293	96344.848459	5361.242291	99262.445910
min	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.000000	0.000000	0.000000	0.000000	0.000000
50%	2.000000	11520.000000	8148.000000	1500.000000	15496.000000
75%	10.000000	28059.500000	31961.000000	3382.500000	37178.000000
max	300.000000	552594.000000	652061.000000	21600.000000	676074.000000

I'm removing the rows & columns, which is not required

In [3]:

```
df1=df[['PARTICULARS', 'UNITS', 'QUANTITY', 'UNIT RATE SUPPLY (Rs.)',
       'supply amount', 'Taxes\n(GST Etc.)\n(Indicate % here)',
       'Total Supply Amount\n(Including Taxes)',
       'UNIT RATE E & C (Rs.) (Basic)','GST Tax\n(Indicate % here)',
       'Errection Amount', 'Total E&C Amount \n(Including Taxes)', 'Total']]
df1.iloc[1:5:,::]
```

Out[3]:

	PARTICULARS	UNITS	QUANTITY	UNIT RATE SUPPLY (Rs.)	supply amount	Taxes\n(GST Etc.)\n(Indicate % here)	Total Supply Amount\n(Including Taxes)	UNI RATE & (Rs (Basic
1	Main Fire Alarm Panel (4 LOOP)\nSupply, Instal...	Nos.	0.0	1 78 200	0.0	18%	0.0	15 00
2	Main Fire Alarm Panel (2 LOOP)\nSupply, Instal...	Nos.	0.0	1 57 000	0.0	18%	0.0	12 00
3	Main Fire Alarm Panel (1 LOOP)\nSupply, Instal...	Nos.	0.0	81 500	0.0	18%	0.0	10 00
4	LCD Repeater Panel \nSupply, Installation, Tes...	Nos.	1.0	46 000	46000.0	18%	54280.0	6 00

In [4]:

```
fas_price = df1['Total'].loc[1:4].sum()
fas_price
```

Out[4]:

```
61360.0
```

In [5]:

```
# df1.iloc[5:26:,::]
```

In [6]:

```
device_details = df1[['PARTICULARS','Total']].loc[6:25]
device_details
```

Out[6]:

	PARTICULARS	Total
6	Addressable Multi Sensor Detector\nSupply, Ins...	76199.0
7	Beam Detector with reflector\nSupply, Installa...	0.0
8	Beam Detector Controller	0.0
9	Optical Smoke Imaging Detector (OSID)\nSupply,...	0.0
10	Heat Detector (Rate of Rise)\nSupply, Installa...	0.0
11	Alarm Initiating Devices	NaN
12	Indoor Manual Call Point\nSupply, Installation...	21028.0
13	Audio & Visual Notification Device	NaN
14	Indoor Hooter cum Strobe\nSupply, Installation...	21358.0
15	Modules	NaN
16	Control Module (For Hooters)\nSupply,Installat...	20414.0
17	Control Relay Module for 3rd Party Integration...	5104.0
18	Monitor Module (For Conventional Detectors)\nS...	0.0
19	Monitor Module (For Fire doors feedback signal...	0.0
20	Relay Module for Future interface with any AHU...	8054.0
21	Input Module for pumproom feedback signal	49324.0
22	Limit switch for fire doors	0.0
23	Fault Isolator Module\nSupply,Installation,Tes...	0.0
24	Response Indicator\nSupply,Installation,Testin...	0.0
25	Power Supply Unit \nSupply & Installation, Tes...	0.0

In [7]:

```
dev_install_Price = df1[['Total']].loc[6:25].sum()
dev_install_Price
```

Out[7]:

```
Total    201481.0
dtype: float64
```

In [8]:

```
cable_details= df1[['PARTICULARS','UNITS','QUANTITY','supply amount','Total']].loc[27:2
cable_details
```

Out[8]:

	PARTICULARS	UNITS	QUANTITY	supply amount	Total
27	2Cx1.5 mm2, Armoured Cable\nSupply & laying o...	Mtr	300.0	28500.0	59118.0

In [9]:

```
cable_install_Price = df1[['Total']].loc[27:27].sum()
cable_install_Price
```

Out[9]:

```
Total    59118.0
dtype: float64
```

In [10]:

```
conduit_details= df1[['PARTICULARS','UNITS','QUANTITY','supply amount','Total']].loc[30
conduit_details
```

Out[10]:

	PARTICULARS	UNITS	QUANTITY	supply amount	Total
30	25 MM METAL CONDUIT\nSupply & laying of 25mm...	Mtr	50.0	14700.0	32096.0

In [11]:

```
conduit_install_Price = df1[['Total']].loc[30:30].sum()
conduit_install_Price
```

Out[11]:

```
Total    32096.0
dtype: float64
```

In [12]:

```
fire_exti_details= df1[['PARTICULARS','UNITS','QUANTITY','supply amount','Total']].loc[
fire_exti_details
```

Out[12]:

	PARTICULARS	UNITS	QUANTITY	supply amount	Total
32	PORTABLE FIRE EXTINGUISHERS	NaN	NaN	NaN	NaN
33	CO2 type Fire Extinguisher 4.5 Kg Capacity mad...	NO	4.0	29460.0	36061.0
34	ABC Dry Powder (MAP 50%) Stored Pressure type ...	NO	8.0	26552.0	33927.0
35	Mechanical Foam Stored Pressure, Portable Type...	NO	4.0	12340.0	15859.0
36	Mechanical Foam type Fire Extinguisher 45 Litr...	NO	1.0	11520.0	14243.0

In [13]:

```
fire_exti_install_Price = df1[['Total']].loc[32:36].sum()
fire_exti_install_Price
```

Out[13]:

```
Total    100090.0
dtype: float64
```

In [14]:

```
mcc_install_Price= df1[['PARTICULARS','UNITS','QUANTITY','supply amount','Total']].loc[
mcc_install_Price
```

Out[14]:

	PARTICULARS	UNITS	QUANTITY	supply amount	Total
37	LT SWITCHGEAR PANELS	NaN	NaN	NaN	NaN
38	FIRE PANEL (MCC PANEL): 415V, 50Hz, short circ...	NO	1.0	552594.0	676074.0
39	Automatic Diesel Engine Control Panel for Stan...	NO	1.0	60000.0	77290.0
40	LT POWER / CONTROL CABLES	NaN	NaN	NaN	NaN
41	3.5 C X 240 sq. mm A2XFY 1100 Voltage grade, M...	M	70.0	70889.0	100004.0
42	3 C X 16 sq. mm A2XFY Power Cables from Fire ...	M	25.0	3452.0	6198.0
43	14 C X 1.5 sq. mm 2XWY Power Cables from Fire...	M	25.0	10933.0	15496.0
44	2 C X 1.5 sq. mm 2XWY	M	250.0	30250.0	60180.0
45	LT CABLE TERMINATION	NaN	NaN	NaN	NaN
46	3.5 C X 240 sq. mm A2XFY	NO	8.0	12152.0	16209.0
47	3 C X 16 sq. mm A2XFY	NO	2.0	767.0	1101.0
48	14 C X 1.5 sq. mm 2XWY	NO	2.0	577.0	888.0
49	2 C X 1.5 sq. mm 2XWY	NO	50.0	6050.0	12036.0

In [15]:

```
mcc_install_Price = df1[['Total']].loc[37:49].sum()
mcc_install_Price
```

Out[15]:

```
Total    965476.0
dtype: float64
```

In [16]:

```
cable_try_details= df1[['PARTICULARS','UNITS','QUANTITY','supply amount','Total']].loc[5
cable_try_details
```

Out[16]:

	PARTICULARS	UNITS	QUANTITY	supply amount	Total
50	CABLE TRAYS	NaN	NaN	NaN	NaN
51	300 mm width (Vertical ladder type cable tray ...	M	10.0	27619.0	35836.0
52	300 mm width (Horizontal ladder type cable tray)	M	15.0	31072.0	41532.0
53	150 mm width (Vertical perforated type cable t...	M	10.0	14070.0	18881.0
54	75 mm width (Vertical perforated type cable tr...	M	20.0	20009.0	27245.0
55	150 mm width (Horizontal perforated type cable...	M	10.0	14070.0	18881.0
56	75 mm width (Horizontal perforated type cable ...	M	8.0	8003.0	10898.0
57	Cable tray supports, supporting frame for moun...	TON	0.3	28770.0	49525.0
58	EARTHING SYSTEM- Earthing system including sup...	NaN	NaN	NaN	NaN
59	50 X 6 mm G.I. STRIP	M	40.0	17799.0	25440.0
60	25 X 6 mm G.I. STRIP	M	30.0	6905.0	10696.0
61	8 SWG GI WIRE	M	20.0	4603.0	7131.0
62	Treated earth pits as per CEIG specific requir...	NO	1.0	25778.0	40802.0

In [17]:

```
cable_try_install_Price = df1[['Total']].loc[50:62].sum()
cable_try_install_Price
```

Out[17]:

```
Total    286867.0
dtype: float64
```

In [18]:

```
switch_details= df1[['PARTICULARS','UNITS','QUANTITY','supply amount','Total']].loc[63:
switch_details
```

Out[18]:

	PARTICULARS	UNITS	QUANTITY	supply amount	Total
63	INSTRUMENTATION	NaN	NaN	NaN	NaN
64	Level Switch for Fire Water Storage Tank	NO	2.0	29154.0	38295.0
65	Pressure Gauge, Bourdon Type, 100 mm Dial and ...	NO	5.0	3376.0	6255.0
66	Pressure Switch, SPDT, Measuring Element - Bel...	NO	9.0	49024.0	61937.0
67	200 NB Flow Meter	NO	1.0	268520.0	318801.0

In [19]:

```
switch_install_Price = df1[['Total']].loc[63:67].sum()
switch_install_Price
```

Out[19]:

```
Total    425288.0
dtype: float64
```

In [20]:

```
south_s1 = df1['Total'].sum()
print('TOTAL ELECTRICAL WORK 0RDER VALUE =',south_s1)
```

```
TOTAL ELECTRICAL WORK 0RDER VALUE = 2131776.0
```

In [21]:

```
a = np.array(['switch_install_Price','cable_try_install_Price', 'mcc_install_Price', 'f
 'cable_install_Price', 'dev_install_Price'])

print(a, type(a))
```

```
['switch_install_Price' 'cable_try_install_Price' 'mcc_install_Price'
 'fire_exti_install_Price' 'fas_price' 'conduit_install_Price'
 'cable_install_Price' 'dev_install_Price'] <class 'numpy.ndarray'>
```

In [22]:

```
# bb = switch_install_Price,cable_try_install_Price,mcc_install_Price,fire_exti_install_
# cable_install_Price,dev_install_Price

det = list(a)
# Sort a details of list for data visualizations
det1=det.sort()

prices = [425288.0,286867.0,965476.0,100090.0,61360.0,32096.0,59118.0,201481.0]
```

In [23]:

```
details = {det:prices for (det,prices) in zip(det,prices)}

# Sort by key
details1 = dict(sorted(details.items()))
details1
```

Out[23]:

```
{'cable_install_Price': 425288.0,
 'cable_try_install_Price': 286867.0,
 'conduit_install_Price': 965476.0,
 'dev_install_Price': 100090.0,
 'fas_price': 61360.0,
 'fire_exti_install_Price': 32096.0,
 'mcc_install_Price': 59118.0,
 'switch_install_Price': 201481.0}
```

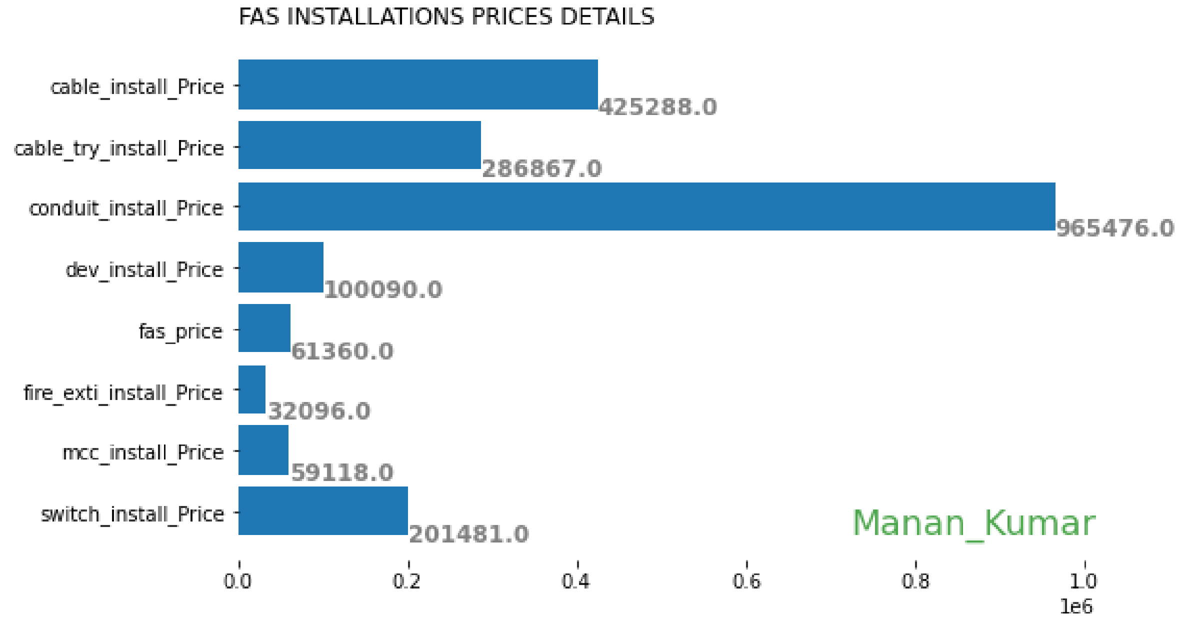
In [24]:

```
# Sort by value
# dict(sorted(prices.items(), key=lambda item: item[1]))
```

FAS INSTALLATIONS PRICES DETAILS of PROJECTS

In [25]:

```
# Figure Size
fig, ax = plt.subplots(figsize =(8, 5))

# Horizontal Bar Plot/ax.barh(name, price)
ax.barh(det, prices)

# # Remove axes splines
for s in ['top', 'bottom', 'left', 'right']:
    ax.spines[s].set_visible(False)

# Show top values
ax.invert_yaxis()

# Add annotation to bars
for i in ax.patches:
    plt.text(i.get_width()+0.8, i.get_y()+0.9,
             str(round((i.get_width()), 5)),
             fontsize = 12, fontweight ='bold',
             color ='grey')

# Add Plot Title
ax.set_title('FAS INSTALLATIONS PRICES DETAILS ',
             loc ='left', )

# Add Text watermark
fig.text(0.9, 0.15, 'Manan_Kumar', fontsize = 18,
         color ='green', ha ='right', va ='bottom',
         alpha = 0.7)

# Show Plot
plt.show()
```



CALCULATTING TIME OF PROJECTS

CONCATENATING DataFrame

In [26]:

```
df2 = df1[['PARTICULARS','UNITS','QUANTITY']].loc[1:4]
df3 = df1[['PARTICULARS','UNITS','QUANTITY']].loc[6:25]
cable_details= df1[['PARTICULARS','UNITS','QUANTITY']].loc[27:27]
conduit_details= df1[['PARTICULARS','UNITS','QUANTITY']].loc[30:30]
fire_exti_details= df1[['PARTICULARS','UNITS','QUANTITY']].loc[32:36]
mcc_install_Price = df1[['PARTICULARS','UNITS','QUANTITY']].loc[37:49]
cable_try_details= df1[['PARTICULARS','UNITS','QUANTITY']].loc[50:62]
switch_details= df1[['PARTICULARS','UNITS','QUANTITY']].loc[63:68]

# concatenating df1 and df2 along rows

vertical_concat = pd.concat([df2, df3, cable_details, conduit_details,fire_exti_details,

# horizontal_concat = pd.concat([df3, df4], axis=1)
```

FAS ADDRESSABLE INT-MODULE INSTALLATIONS TIME

In [27]:

```
vertical_concat.loc[5:25]
```

Out[27]:

	PARTICULARS	UNITS	QUANTITY
6	Addressable Multi Sensor Detector\nSupply, Ins...	Nos	21.0
7	Beam Detector with reflector\nSupply, Installa...	Nos	0.0
8	Beam Detector Controller	Nos	0.0
9	Optical Smoke Imaging Detector (OSID)\nSupply,...	QRO	0.0
10	Heat Detector (Rate of Rise)\nSupply, Installa...	QRO	0.0
11	Alarm Initiating Devices	NaN	NaN
12	Indoor Manual Call Point\nSupply, Installation...	Nos	4.0
13	Audio & Visual Notification Device	NaN	NaN
14	Indoor Hooter cum Strobe\nSupply, Installation...	Nos	4.0
15	Modules	NaN	NaN
16	Control Module (For Hooters)\nSupply,Installat...	Nos	4.0
17	Control Relay Module for 3rd Party Integration...	Nos.	1.0
18	Monitor Module (For Conventional Detectors)\nS...	Nos	0.0
19	Monitor Module (For Fire doors feedback signal...	Nos	0.0
20	Relay Module for Future interface with any AHU...	Nos	1.0
21	Input Module for pumproom feedback signal	Nos	8.0
22	Limit switch for fire doors	Nos	0.0
23	Fault Isolator Module\nSupply,Installation,Tes...	Nos	0.0
24	Response Indicator\nSupply,Installation,Testin...	Nos.	0.0
25	Power Supply Unit \nSupply & Installation, Tes...	Nos	0.0

In [28]:

```
DEVICE = vertical_concat['QUANTITY'].loc[5:25].sum()/5
print("module_installation_time = ",DEVICE, "Days")
```

```
module_installation_time =  8.6 Days
```

REPEATER PANEL INSTALLATION

In [29]:

```
rep_panel_time = vertical_concat[['PARTICULARS','UNITS','QUANTITY']].loc[1:4]
rep_panel_time
```

Out[29]:

	PARTICULARS	UNITS	QUANTITY
1	Main Fire Alarm Panel (4 LOOP)\nSupply, Instal...	Nos.	0.0
2	Main Fire Alarm Panel (2 LOOP)\nSupply, Instal...	Nos.	0.0
3	Main Fire Alarm Panel (1 LOOP)\nSupply, Instal...	Nos.	0.0
4	LCD Repeater Panel \nSupply, Installation, Tes...	Nos.	1.0

In [30]:

```
PANEL = rep_panel_time['QUANTITY'].sum()
print("rp_installation_time = ",PANEL, "Days")
```

```
rp_installation_time =  1.0 Days
```

CABLE INSTALLATIONS TIME

In [31]:

```
vertical_concat.iloc[24:26]
```

Out[31]:

	PARTICULARS	UNITS	QUANTITY
27	2Cx1.5 mm2, Armoured Cable\nSupply & laying o...	Mtr	300.0
30	25 MM METAL CONDUIT\nSupply & laying of 25mm...	Mtr	50.0

In [32]:

```
CABLE_LAYING  = vertical_concat['QUANTITY'].iloc[24:26].sum()/100
print('CABLE_LAYING TIME =',CABLE_LAYING,'days')
```

```
CABLE_LAYING TIME = 3.5 days
```

PORTABLE FIRE EXTINGUISHERS installation time

In [33]:

```
vertical_concat.iloc[26:31]
```

Out[33]:

	PARTICULARS	UNITS	QUANTITY
32	PORTABLE FIRE EXTINGUISHERS	NaN	NaN
33	CO2 type Fire Extinguisher 4.5 Kg Capacity mad...	NO	4.0
34	ABC Dry Powder (MAP 50%) Stored Pressure type ...	NO	8.0
35	Mechanical Foam Stored Pressure, Portable Type...	NO	4.0
36	Mechanical Foam type Fire Extinguisher 45 Litr...	NO	1.0

In [34]:

```
FIRE_EXT_CYL = vertical_concat['QUANTITY'].iloc[26:31].sum()/30
if FIRE_EXT_CYL > 0.5 and FIRE_EXT_CYL< 0.9:
    FIRE_EXT_CYL = round(FIRE_EXT_CYL)
else:
    FIRE_EXT_CYL

print('CYL_INSTALLATION TIME =',FIRE_EXT_CYL,'days')
```

```
CYL_INSTALLATION TIME = 1 days
```

LT Panel installation time

In [35]:

```
vertical_concat.iloc[31:34]
```

Out[35]:

	PARTICULARS	UNITS	QUANTITY
37	LT SWITCHGEAR PANELS	NaN	NaN
38	FIRE PANEL (MCC PANEL): 415V, 50Hz, short circ...	NO	1.0
39	Automatic Diesel Engine Control Panel for Stan...	NO	1.0

In [36]:

```
LT_PANEL_MCC = vertical_concat['QUANTITY'].iloc[31:34].sum()*2
print('LT PANELS INSTALLATION TIME =',LT_PANEL_MCC,'days')
```

```
LT PANELS INSTALLATION TIME = 4.0 days
```

LT POWER / CONTROL CABLES laaying & connection time

In [37]:

```
vertical_concat.iloc[34:39]
```

Out[37]:

	PARTICULARS	UNITS	QUANTITY
40	LT POWER / CONTROL CABLES	NaN	NaN
41	3.5 C X 240 sq. mm A2XFY 1100 Voltage grade, M...	M	70.0
42	3 C X 16 sq. mm A2XFY Power Cables from Fire ...	M	25.0
43	14 C X 1.5 sq. mm 2XWY Power Cables from Fire...	M	25.0
44	2 C X 1.5 sq. mm 2XWY	M	250.0

In [38]:

```
CONDUIT_LAYING = vertical_concat['QUANTITY'].iloc[34:39].sum()/100
print('CABLE INSTALLATION TIME =',CONDUIT_LAYING ,'days')
```

```
CABLE INSTALLATION TIME = 3.7 days
```

LT CABLE TERMINATION time

In [39]:

```
vertical_concat.iloc[39:44]
```

Out[39]:

	PARTICULARS	UNITS	QUANTITY
45	LT CABLE TERMINATION	NaN	NaN
46	3.5 C X 240 sq. mm A2XFY	NO	8.0
47	3 C X 16 sq. mm A2XFY	NO	2.0
48	14 C X 1.5 sq. mm 2XWY	NO	2.0
49	2 C X 1.5 sq. mm 2XWY	NO	50.0

In [40]:

```
ltcab = vertical_concat['QUANTITY'].iloc[39:44].sum()
LT_CABLE_INS_T = (ltcab*2)/50
print('LT CABLE INSTALLATION TIME =',LT_CABLE_INS_T,'days')
```

```
LT CABLE INSTALLATION TIME = 2.48 days
```

CABLE TRAYS INSTALLATIONS TIME

In [41]:

```
vertical_concat.iloc[44:51]
```

Out[41]:

	PARTICULARS	UNITS	QUANTITY
50	CABLE TRAYS	NaN	NaN
51	300 mm width (Vertical ladder type cable tray ...	M	10.0
52	300 mm width (Horizontal ladder type cable tray)	M	15.0
53	150 mm width (Vertical perforated type cable t...	M	10.0
54	75 mm width (Vertical perforated type cable tr...	M	20.0
55	150 mm width (Horizontal perforated type cable...	M	10.0
56	75 mm width (Horizontal perforated type cable ...	M	8.0

In [42]:

```
tray = vertical_concat['QUANTITY'].iloc[44:51].sum()
TRAY_FIXING= tray/50
print('CABLE TRAYS INSTALLATION TIME =',TRAY_FIXING,'days')
```

```
CABLE TRAYS INSTALLATION TIME = 1.46 days
```

Cable tray supports fabrication with paint

In [43]:

```
vertical_concat.iloc[51:52]
```

Out[43]:

	PARTICULARS	UNITS	QUANTITY
57	Cable tray supports, supporting frame for moun...	TON	0.3

In [44]:

```
sup = vertical_concat['QUANTITY'].iloc[51:52].sum()*1000
MS_SUPPORT= (sup/4)/30
print('TRAY SUPPORT FABRICATION TIME =',MS_SUPPORT,'days')
```

```
TRAY SUPPORT FABRICATION TIME = 2.5 days
```

EARTHING INSTALLATION IN PUMP ROOM

In [45]:

```
vertical_concat.iloc[52:56]
```

Out[45]:

	PARTICULARS	UNITS	QUANTITY
58	EARTHING SYSTEM- Earthing system including sup...	NaN	NaN
59	50 X 6 mm G.I. STRIP	M	40.0
60	25 X 6 mm G.I. STRIP	M	30.0
61	8 SWG GI WIRE	M	20.0

In [46]:

```
erth = vertical_concat['QUANTITY'].iloc[52:56].sum()
EARTHING_CABLE = erth/50
if EARTHING_CABLE > 0.5 and EARTHING_CABLE < 0.9:
    EARTHING_CABLE = round(EARTHING_CABLE)
else:
    EARTHING_CABLE

print('EARTHING INSTALLATION TIME =',EARTHING_CABLE,'days')
```

```
EARTHING INSTALLATION TIME = 1.8 days
```

EARTH PITS INSTALLATION TIME

In [47]:

```
vertical_concat.iloc[56:57]
```

Out[47]:

	PARTICULARS	UNITS	QUANTITY
62	Treated earth pits as per CEIG specific requir...	NO	1.0

In [48]:

```
EARTH_PITS = vertical_concat['QUANTITY'].iloc[56:57].sum()*1.5
print('EARTH-PITS INSTALLATION TIME =',EARTH_PITS,'days')
```

```
EARTH-PITS INSTALLATION TIME = 1.5 days
```

LEVEL SWITCH & PRESSURE SWITCH INSTALLATIONS TIME

In [49]:

```
vertical_concat.iloc[57:68]
```

Out[49]:

	PARTICULARS	UNITS	QUANTITY
63	INSTRUMENTATION	NaN	NaN
64	Level Switch for Fire Water Storage Tank	NO	2.0
65	Pressure Gauge, Bourdon Type, 100 mm Dial and ...	NO	5.0
66	Pressure Switch, SPDT, Measuring Element - Bel...	NO	9.0
67	200 NB Flow Meter	NO	1.0

In [50]:

```
LSPS_SWITCH = vertical_concat['QUANTITY'].iloc[57:68].sum()/10
print('LIMIT & PRESSURE SW =',LSPS_SWITCH,'days')
```

```
LIMIT & PRESSURE SW = 1.7 days
```

TOTAL REQUIRED INSTALLATION TIME

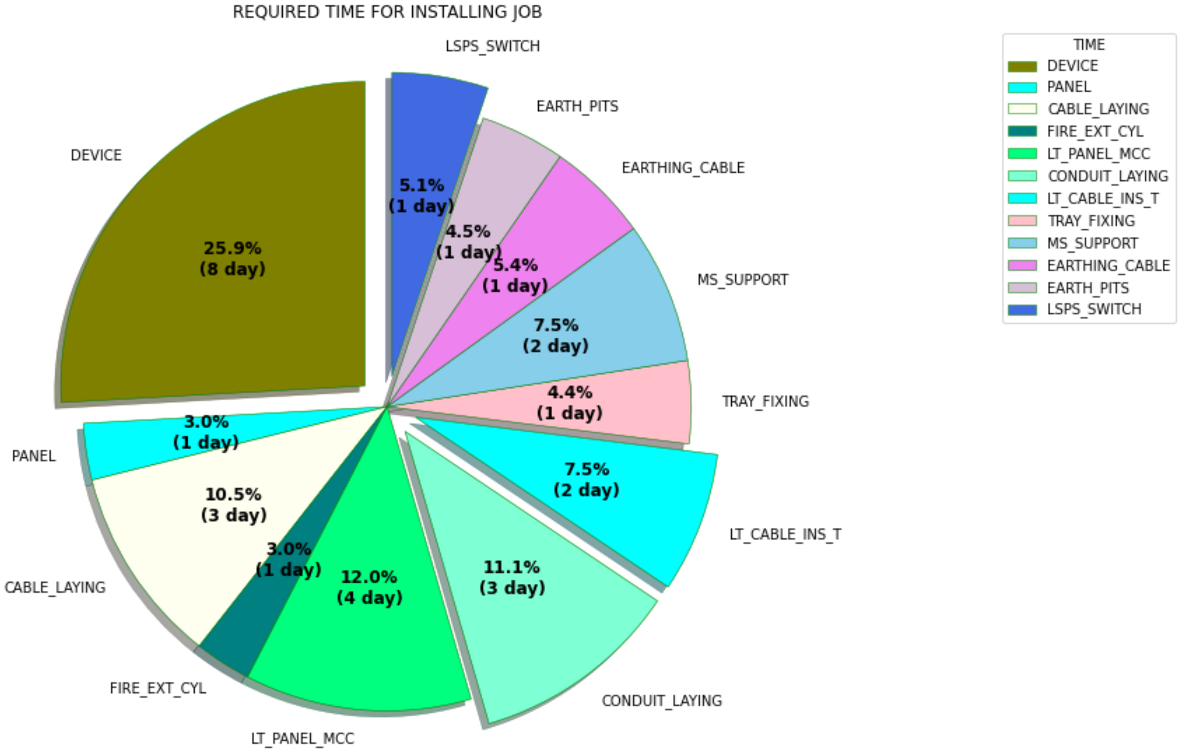
In [51]:

```
days = [DEVICE,PANEL,CABLE_LAYING,FIRE_EXT_CYL,LT_PANEL_MCC,CONDUIT_LAYING,LT_CABLE_INS_
        EARTHING_CABLE,EARTH_PITS,LSPS_SWITCH]
d = 0
for i in days:
    d = d+i
print('TOTAL INSTALLATIONS TIME =', d,'Days')
```

```
TOTAL INSTALLATIONS TIME = 33.24 Days
```

In [52]:

```
# # Creating dataset
time = ['DEVICE','PANEL','CABLE_LAYING','FIRE_EXT_CYL','LT_PANEL_MCC','CONDUIT_LAYING',
        'TRAY_FIXING','MS_SUPPORT','EARTHING_CABLE','EARTH_PITS','LSPS_SWITCH']

# Creating explode data
explode = (0.1, 0.0, 0.0, 0.0, 0.0, 0.1,0.1, 0.0, 0.0, 0.0, 0.0, 0.1)

# Creating color parameters
colors = ( "olive", "cyan", "ivory","teal", "springgreen", "aquamarine","aqua", "pink",

# Wedge properties
wp = { 'linewidth' : 0.5, 'edgecolor' : "green" }

# Creating autocpt arguments
def func(pct, allvalues):
    absolute = int(pct / 100.*np.sum(allvalues))
    return "{:.1f}%\n({:d} day)".format(pct, absolute)

# Creating plot
fig, ax = plt.subplots(figsize =(20, 10))
wedges, texts, autotexts = ax.pie(days,
                                  autopct = lambda pct: func(pct, days),
                                  explode = explode,
                                  labels = time,
                                  shadow = True,
                                  colors = colors,
                                  startangle = 90,
                                  wedgeprops = wp,
                                  textprops = dict(color ="black"))

# Adding legend /magenta
ax.legend(wedges, time,
          title ="TIME",
          loc ="center left",
          bbox_to_anchor =(1.3, 0.3, 0.5, 1))

plt.setp(autotexts, size = 12, weight ="bold")
ax.set_title("REQUIRED TIME FOR INSTALLING JOB")

# show plot
plt.show()
```



ACCESSORIESE OF FITING MATERIALS

In [101]:

```
# for panel & devices accessories

t1=vertical_concat.loc[1:4]
t2=vertical_concat.loc[5:25]
tt = pd.concat([t1,t2],axis=0)
t = tt['QUANTITY'].sum()
# print('PG Gland = ',t*5)
# print('35x8 screw= ',t*3)
# print('6mm PVC Fixers= ',t*2)
```

In [102]:

```
# for cable saddeling required accessories

t3=vertical_concat.loc[27:27]
t4=vertical_concat.loc[30:30]
tt1 = pd.concat([t3,t4],axis=0)
t2 = tt1['QUANTITY'].sum()
# print('saddle 12mm = ',t2*3)
# print('35x8 screw= ',t2*3)
# print('6mm PVC Fixers= ',t2*3)
```

In [103]:

```
t6=vertical_concat.loc[40:44]
t3 = t6['QUANTITY'].sum()
# print('SS Cable tie=',t3)
# print('PVC Cable tie=',t3)
# print('240 Sqr. mm SS Gland=', 6 ,'Nos.')
# print('16 Sqr. mm SS Gland=', 6 ,'Nos.')
```

In [104]:

```
#t7=vertical_concat[['PARTICULARS','UNITS','QUANTITY']].loc[50:61]
t7=vertical_concat.loc[50:61]
t4 = t7['QUANTITY'].sum()
# print('For Cable tray nut/bolts 25mm =',t4*.5)
```

THIS TIME CALCULATED BASED FOR 2 ELECTRICIANS + 2 HELPER = TOTAL 4 PERSONS

In [105]:

```
print('TOTAL ELECTRICAL WORK 0RDER VALUE =',south_s1)
print('TOTAL INSTALLATIONS TIME =', d,'Days')
print('PG Gland = ',t*5)
print('35x8 screw= ',t*3)
print('6mm PVC Fixers= ',t*2)
print('saddle 12mm = ',t2*3)
print('35x8 screw= ',t2*3)
print('6mm PVC Fixers= ',t2*3)
print('SS Cable tie=',t3)
print('PVC Cable tie=',t3)
print('240 Sqr. mm SS Gland=', 6 ,'Nos.')
print('16 Sqr. mm SS Gland=', 6 ,'Nos.')
print('For Cable tray nut/bolts 25mm =',t4*.5)
```

```
TOTAL INSTALLATIONS TIME = 33.24 Days
TOTAL ELECTRICAL WORK 0RDER VALUE = 2131776.0
PG Gland =  220.0
35x8 screw=  132.0
6mm PVC Fixers=  88.0
saddle 12mm =  1050.0
35x8 screw=  1050.0
6mm PVC Fixers=  1050.0
SS Cable tie= 370.0
PVC Cable tie= 370.0
240 Sqr. mm SS Gland= 6 Nos.
16 Sqr. mm SS Gland= 6 Nos.
For Cable tray nut/bolts 25mm = 81.65
```

In [ ]:

```

```

In [ ]:

```

```