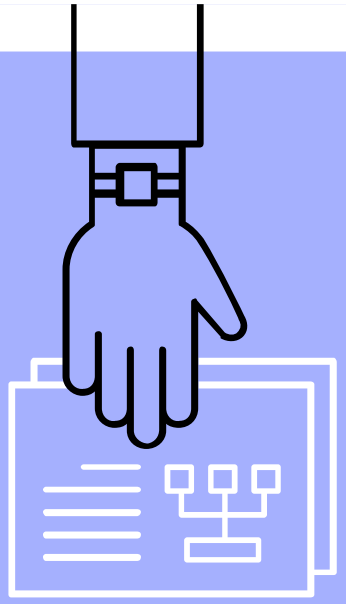
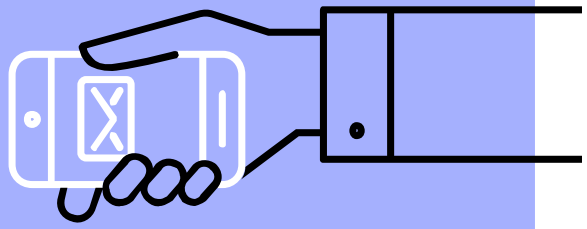
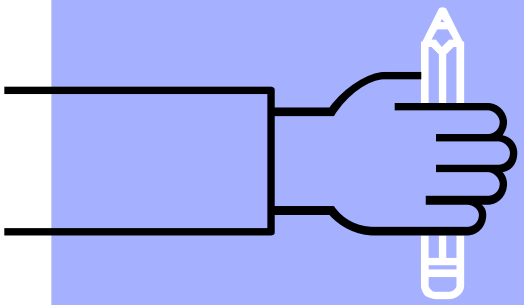
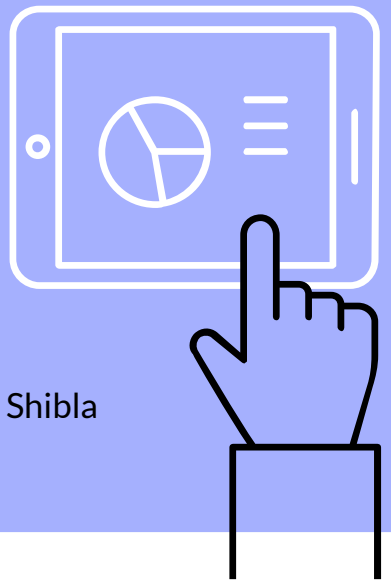




Simulation Modeling, Analysis, & Outsourcing

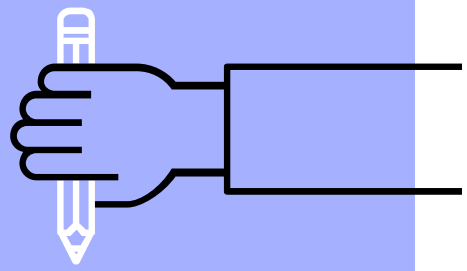




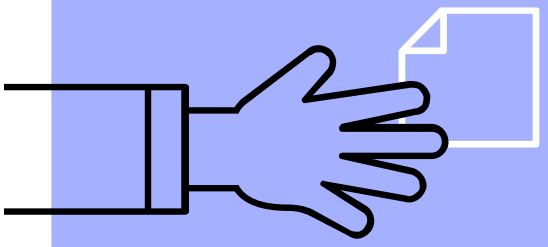
IDS 400, Final Project

Mohammad Abdelmajid, Omar Salka, Harshal Patel, Adam Shibla

Introduction



- Maintenance Facility currently providing inspection and repair services to Enterprise Rent-A-Car.
- Interested in increasing market share with an additional partnership

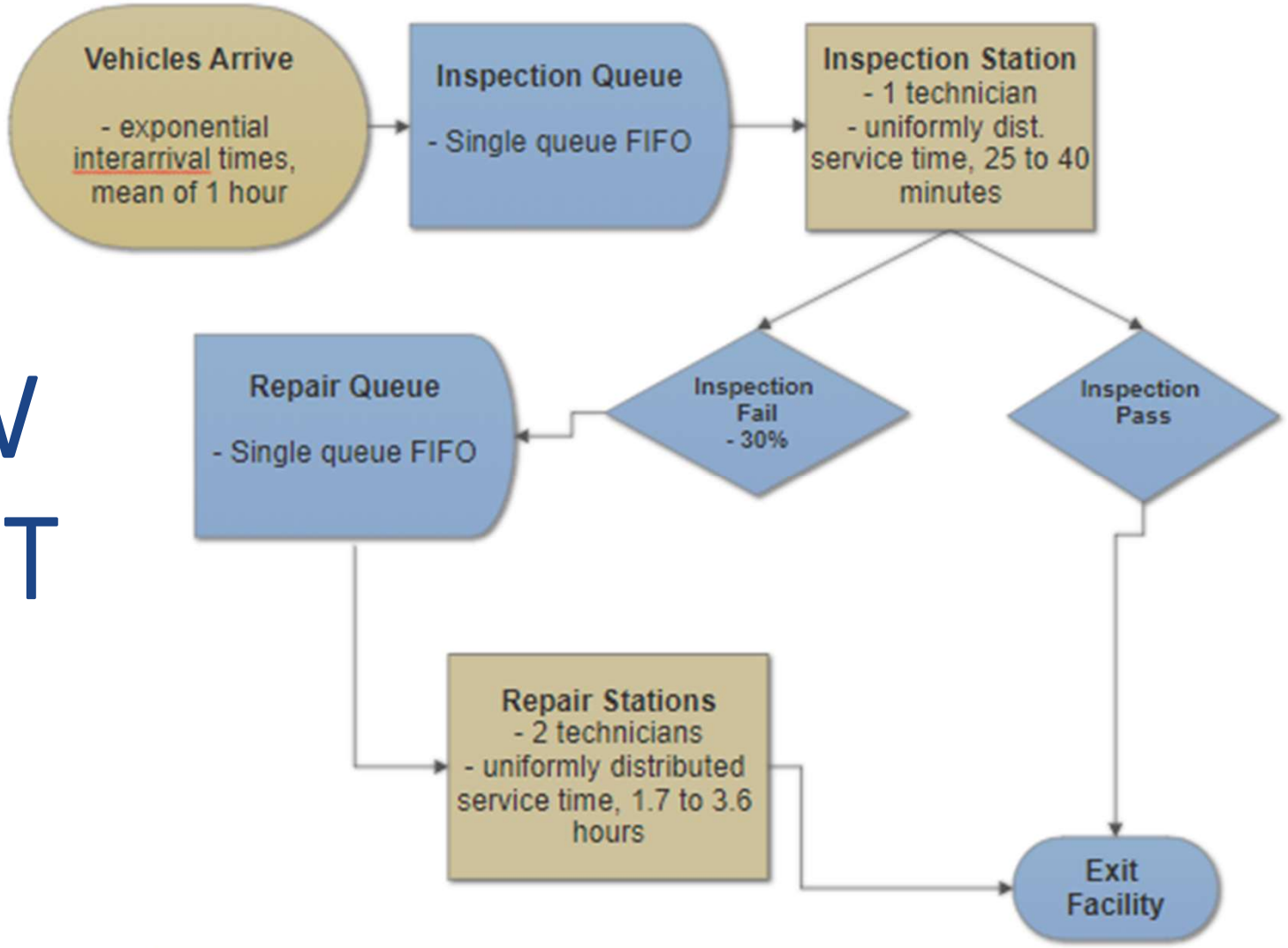


* Inspired by 420 simulation class

Our Project Outline

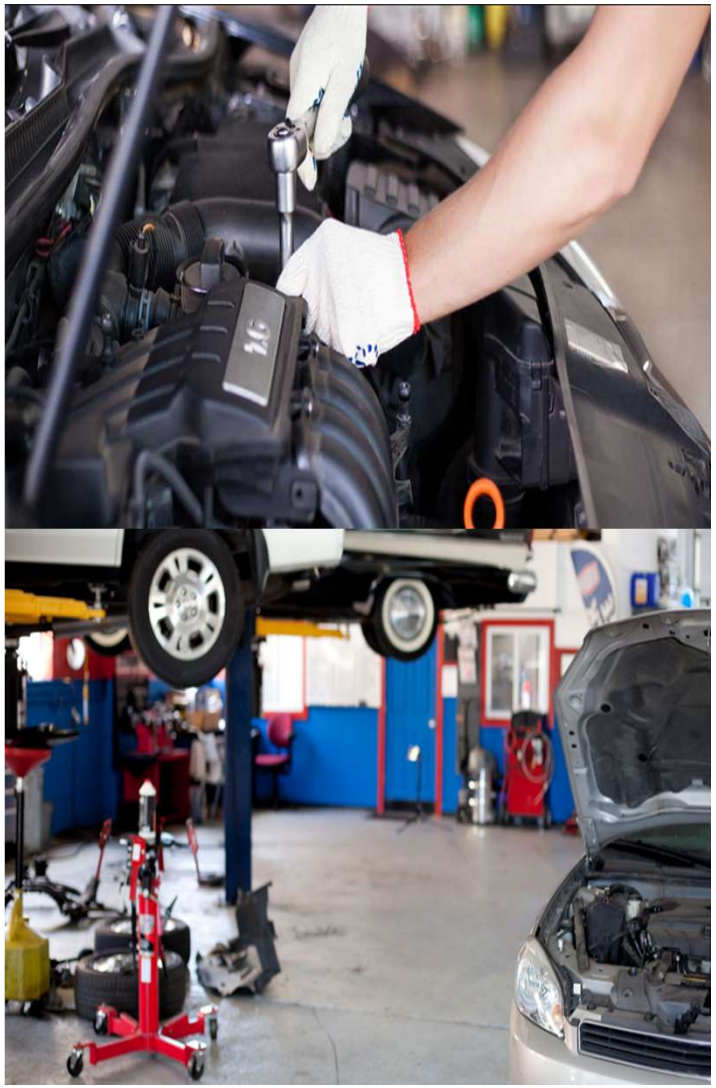
- Model the maintenance facility using Python to examine current operational efficiency
- Simulate the operations for 1 week, 5 times over
- Examine current business model & explore changes for improvement
- Provide analysis for a revised business model & improvements

FLOW CHART


Vehicles Arrive
- exponential interarrival times, mean of 1 hour
Inspection Queue
- Single queue FIFO
Inspection Station
- 1 technician
- uniformly dist. service time, 25 to 40 minutes
Inspection Fail
- 30%
Inspection Pass
Repair Queue
- Single queue FIFO
Repair Stations
- 2 technicians
- uniformly distributed service time, 1.7 to 3.6 hours
Exit Facility

Key Measurements

- Average delay of each queue, in hours
- Average length of each queue, in vehicles
- Utilization of both the inspection stations and repair stations
- Service charge revenue
- Capacity for throughput of vehicles
- Wages wasted on idle workers



About the code:

Libraries Used:

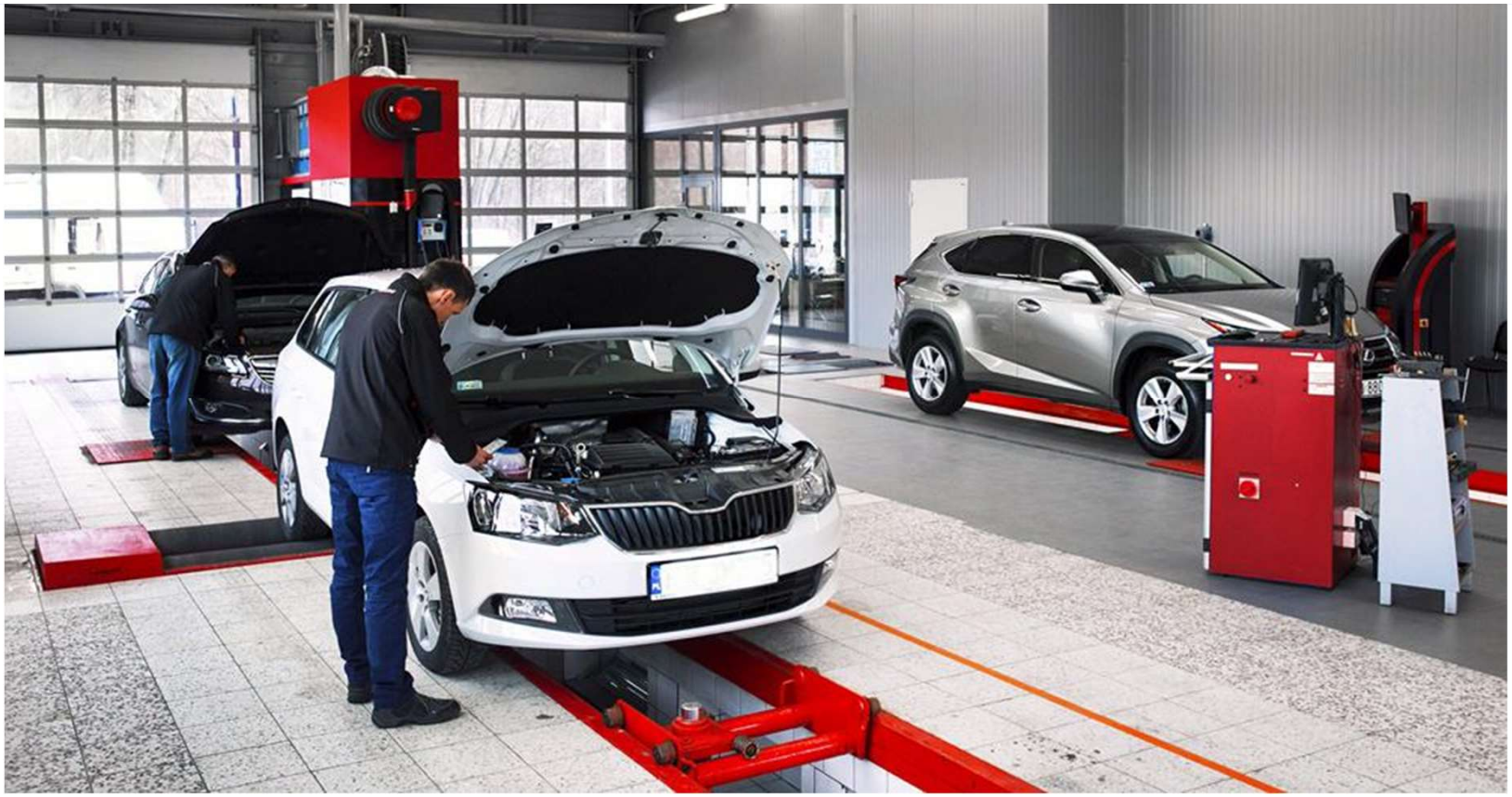
- ➔ Pandas
- ➔ Numpy

Number generators (controlled randomness):

```
#generate service time
return np.random.uniform(1.7*0.75, 3.6*0.75)
```

```
#discrete random number generator
next_step = ["Exit", "Repair Station"]
probabilities = [0.7, 0.3]
np.random.choice(next_step, p=probabilities)
```

```
#Exponential Random Number Generator
return np.random.exponential(0.5)
```

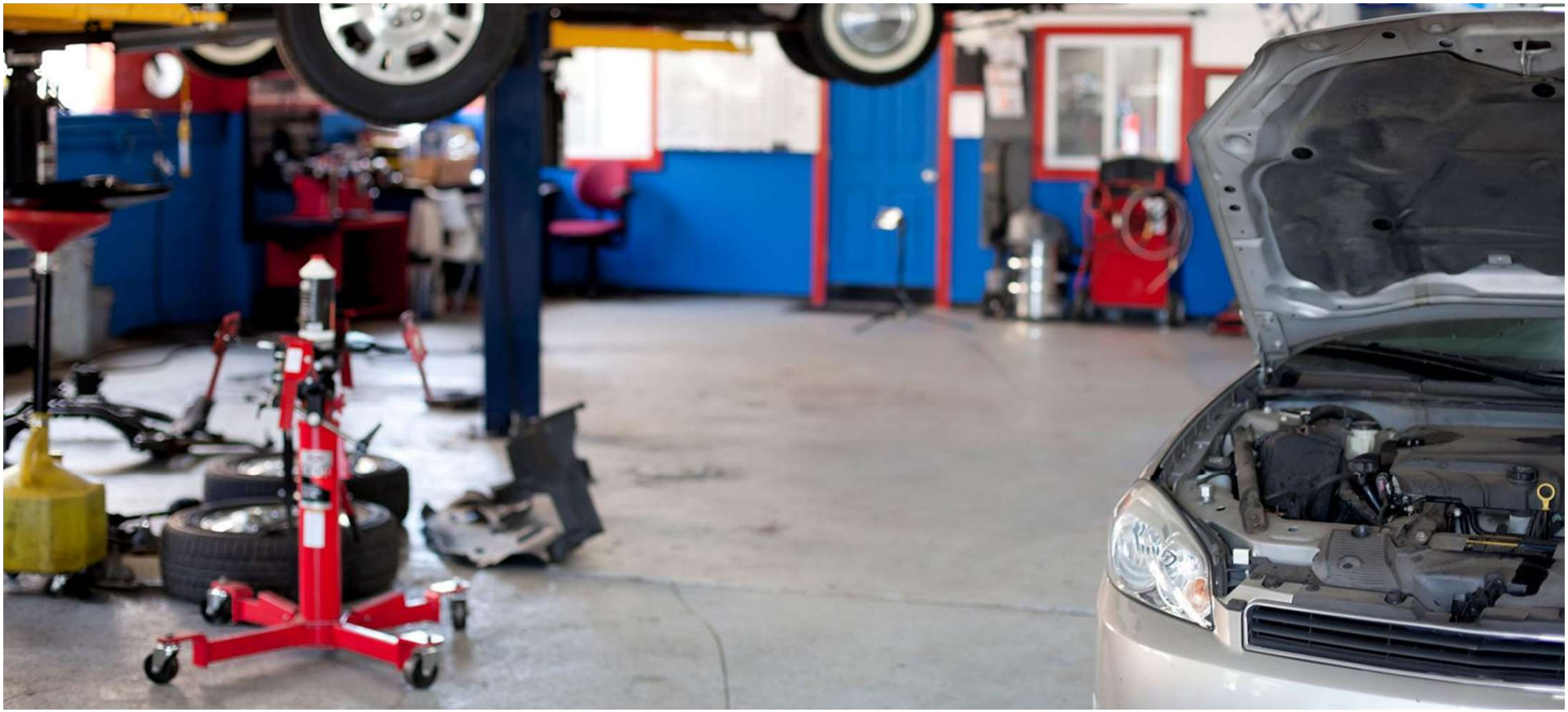


Inspection Station

The Data:

Inspection Station

Index	event_type	clock	interarrival_time	service_time	system_state	waiting_in_line	next_arrival	next_departure	insys	time_btw_events	next_stop
0	Arrival	0.627965	0.393601	0.425518	1	0	1.02157	1.05348	1	0.393601	
1	Arrival	1.02157	0.275524	0.0319176	1	1	1.29709		2	0.0319176	
2	Departure	1.05348			1	0		1.44803	1	0.243607	Exit
3	Arrival	1.29709	1.11176	0.150941	1	1	2.40885		2	0.150941	
4	Departure	1.44803			1	0		1.83243	1	0.384395	Repair Station
5	Departure	1.83243			0	0		inf	0	0.576426	Repair Station
6	Arrival	2.40885	0.419716	0.411668	1	0	2.82857	2.82052	1	0.411668	
7	Departure	2.82052			0	0		inf	0	0.00804863	Repair Station
8	Arrival	2.82857	0.0455805	0.325819	1	0	2.87415	3.15439	1	0.0455805	
9	Arrival	2.87415	0.0102128	0.280239	1	1	2.88436		2	0.0102128	
10	Arrival	2.88436	0.893744	0.270026	1	2	3.77811		3	0.270026	
11	Departure	3.15439			1	1		3.63002	2	0.475627	Repair Station
12	Departure	3.63002			1	0		4.09236	1	0.148091	Repair Station
13	Arrival	3.77811	0.309465	0.314252	1	1	4.08757		2	0.309465	
14	Arrival	4.08757	0.758268	0.00478689	1	2	4.84584		3	0.00478689	
15	Departure	4.09236			1	1		4.52484	2	0.432485	Exit
16	Departure	4.52484			1	0		5.01447	1	0.320996	Exit
17	Arrival	4.84584	0.368914	0.16863	1	1	5.21475		2	0.16863	
18	Departure	5.01447			1	0		5.37657	1	0.200284	Exit
19	Arrival	5.21475	0.744127	0.16182	1	1	5.95888		2	0.16182	
20	Departure	5.37657			1	0		5.79565	1	0.419081	Exit
21	Departure	5.79565			0	0		inf	0	0.163226	Exit
22	Arrival	5.95888	0.473498	0.428307	1	0	6.43238	6.38719	1	0.428307	
23	Departure	6.38719			0	0		inf	0	0.0451917	Exit



Repair Station

The Data:

Repair Stations

Index	event_type	clock	interarrival_time	service_time	system_state	waiting_in_line	next_arrival	next_departure	insys	time_btw_events	repairs	service_charge
0	Arrival	0	0.384395	1.99557	1	0	0.384395	1.99557	1	0.384395	Oil Change, A/C	537.5
1	Arrival	0.384395	0.988094	1.61117	1	0	1.37249		2	0.988094	Oil Change, Brakes(4 axle…	637.5
2	Arrival	1.37249	0.333868	0.623081	1	1	1.70636		3	0.333868	Oil Change	37.5
3	Arrival	1.70636	0.475627	0.289213	1	2	2.18198		4	0.289213	Leaks, Oil Change, Compr…	912.5
4	Departure	1.99557			1	1		3.55384	3	0.186414		
5	Arrival	2.18198	8.29769	1.37186	1	2	10.4797		4	1.37186	Brakes(4 axles)	600
6	Departure	3.55384			1	1		4.91799	3	1.36414		
7	Departure	4.91799			1	0		6.89618	2	1.97819		
8	Departure	6.89618			1	0		9.1041	1	2.20793		
9	Departure	9.1041			0	0		inf	0	1.37557		
10	Arrival	10.4797	1.32194	1.71988	1	0	11.8016	12.1996	1	1.32194	Oil Change	37.5
11	Arrival	11.8016	1.54082	0.397942	1	0	13.3424		2	0.397942	Leaks, Brakes(4 axle…	725
12	Departure	12.1996			1	0		13.7145	1	1.14288		
13	Arrival	13.3424	8.36925	0.372056	1	0	21.7117		2	0.372056	Leaks, Brakes(4 axle…	725
14	Departure	13.7145			1	0		15.4068	1	1.69228		
15	Departure	15.4068			0	0		inf	0	6.30491		
16	Arrival	21.7117	0.463661	2.22126	1	0	22.1753	23.9329	1	0.463661	Brakes(4 axles)	600
17	Arrival	22.1753	0.531385	1.7576	1	0	22.7067		2	0.531385	Leaks, Oil Change, Paint	728.5
18	Arrival	22.7067	3.68233	1.22622	1	1	26.3891		3	1.22622	Leaks, A/C, Brakes(4 axle…	1225
19	Departure	23.9329			1	0		25.6914	2	1.75849		
20	Departure	25.6914			1	0		27.3009	1	0.69762		
21	Arrival	26.3891	2.24531	0.911885	1	0	28.6344		2	0.911885	Leaks	125
22	Departure	27.3009			1	0		28.7286	1	1.33342		
23	Arrival	28.6344	2.39859	0.0942	1	0	31.0329		2	0.0942	Leaks, Oil Change, A/C	662.5

How we assigned repairs and repair costs based on probability

```
def generate_repair_n_cost(self):
    #input for repair random generator
    repair_list = ["Leaks", "Compressor Failure", "System Contamination",
                   "Oil Change", "Tires", "Paint", "A/C", "Brakes(4 axles)"]
    prob = [0.20, 0.02, 0.02, 0.35, 0.05, 0.08, 0.10, 0.18]

    #costs of repairs
    repair_cost = [125, 750, 1150, 37.5, 400, 566, 500, 600]

    #create an array of randomly generated repairs
    list_r = []
    for i in range(np.random.randint(1, 4)):
        list_r.append(np.random.choice(repair_list, p=prob))

    #to remove duplicates
    new_list= set(list_r)

    #healps with the search for associated costs
    repair_dictionnary = dict(zip(repair_list, repair_cost))


    #search for associated costs
    accumulated_cost = 0
    for i in repair_list:
        if i in new_list:
            accumulated_cost += repair_dictionnary[i]

    self.repairs.append(str(new_list).replace("{", "").replace("}", "").replace("'", ""))
    self.cost.append(accumulated_cost)
    self.total_service_charge += accumulated_cost
```

Results - Current State:

Original State

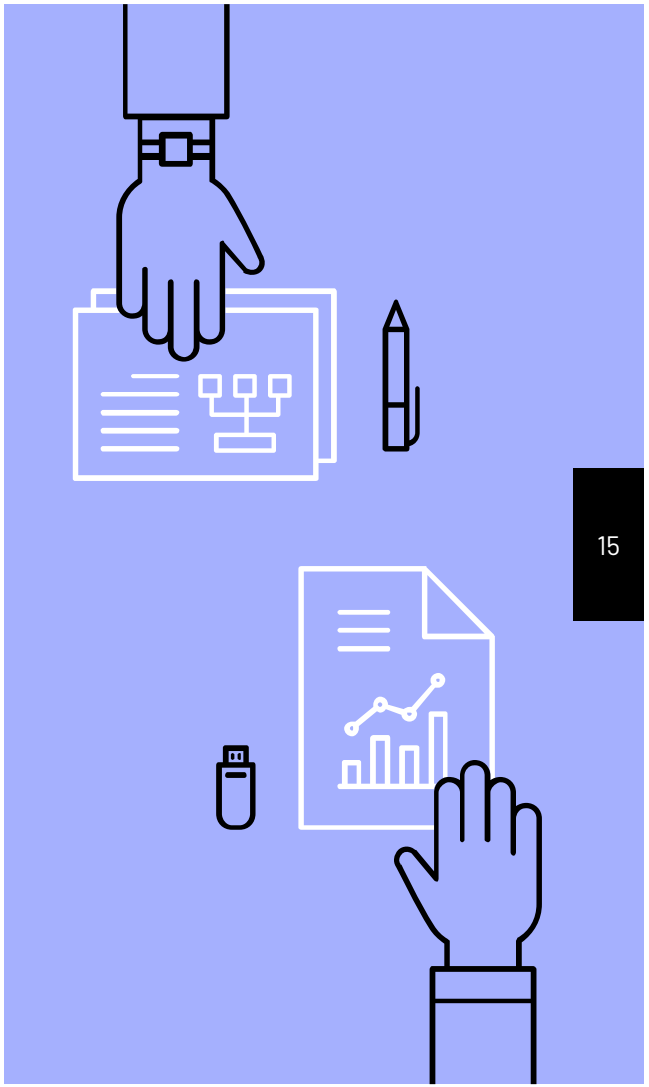
	InspectionAverage	RepairAverage	
Average delay of queue (in hours)	0.314	0.223	
Average length of queue (in cars)	0.598	0.588	
Utilization of staff	51.44%	37.40%	
Total Cars in System (arrivals)	152	25	
Total Service Charge	$ 4,602	$ 13,541	**$ 18,143**
Wasted wages (idle time paid)	$ 1,528	$ 1,127	**$ 2,655**

Adding Partnership

- Currently servicing Enterprise cars
- Propose to Avis that we will maintain their cars
- Increase capacity for additional incoming vehicles
- Original time is 1 hour

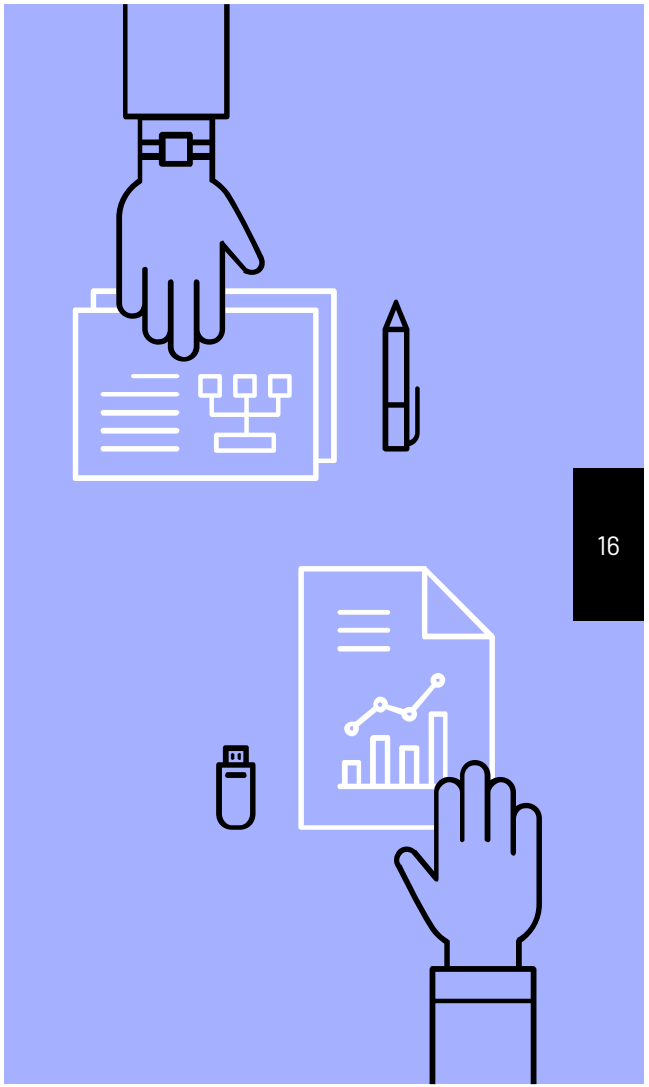
Exploring new opportunities

- We need to invest in new equipment to meet Avis' volume needs
- We were underutilized at the moment
- Need to improve both our inspection and repair service time
- Adding more partnerships means our mean inter-arrival will decrease in half (to 30 mins)



Business Decisions

- Meet the 80%utilization rate for our company standards
- Decrease both inspection and repair service time by 25% by upgrading equipment



Code Change

Current State

```
def generate_interarrival(self):
    #Exponential Random Number Generator
    return np.random.exponential(1)
```

```
def generate_service_time(self):
    #Uniform Random Number Generator
    return np.random.uniform(0.416667, 0.666667 )
```

```
def generate_service_time(self):
    #generate service time
    return np.random.uniform(1.7, 3.6)
```

Changed State

```
def generate_interarrival(self):
    #Exponential Random Number Generator
    return np.random.exponential(0.5)
```

```
def generate_service_time(self):
    return np.random.uniform(0.31, 0.5)
```

```
def generate_service_time(self):
    return np.random.uniform(1.27,2.7)
```

Improvement Analysis

Original State

	Inspection Average	Repair Average	
Average delay of queue (in hours)	0.314	0.223	
Average length of queue (in cars)	0.598	0.588	
Utilization of staff	51.44%	37.40%	
Total Cars in System (arrivals)	152	25	
Total Service Charge	$ 4,602	$ 13,541	$ 18,143
Wasted wages (idle time paid)	$ 1,528	$ 1,127	$ 2,655

Recommended State

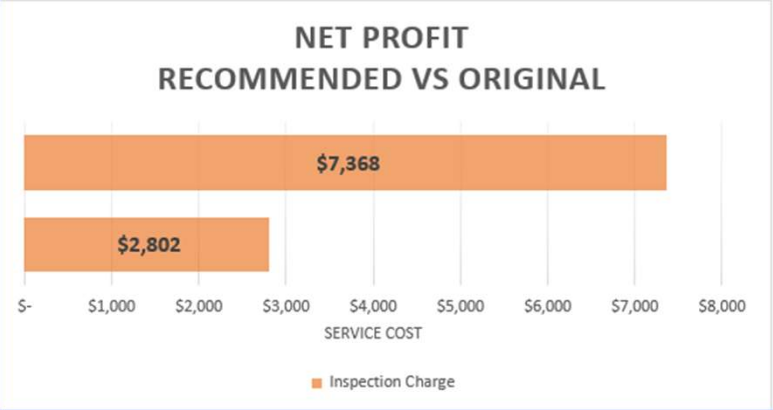
	Inspection Average	Repair Average	
Average delay of queue (in hours)	1.683	1.035	
Average length of queue (in cars)	2.054	2.006	
Utilization of staff	80.89%	49.12%	
Total Cars in System (arrivals)	329	46	
Total Service Charge	$ 9,678	$ 27,032	$ 36,710
Wasted wages (idle time paid)	$ 604	$ 718	$ 1,322

Financial Analysis

Original Avg Inspection		Inspection Charge
152	$	4,602
Total After Exp	$	2,802

Recommended State Avg Inspection		Inspection Charge
329	$	9,678
Total After Exp	$	7,368

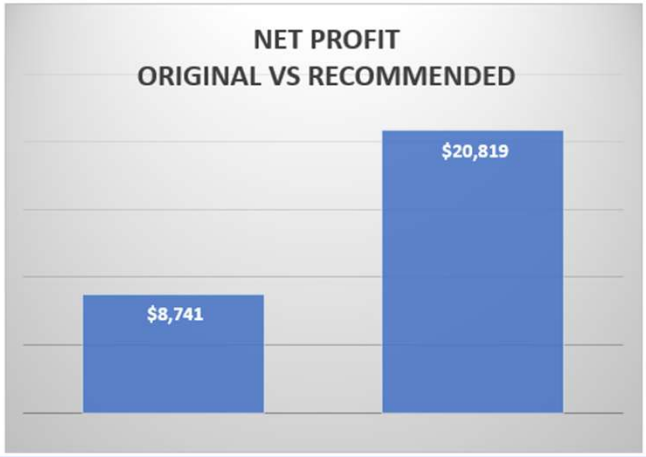
Arrival Difference		Increased Inspection Profit
177	**$**	**4,566**
(116% Increase)		(181% Increase)



Original Avg Repair		Repair Charge
25	$	13,541
Total After Exp	$	8,741

Recommended State Avg Repair		Repair Charge
46	$	27,032
Total After Exp	$	20,819

Repair Differences		Increased Repair Profit
21	**$**	**12,078**
(84% Increase)		(154% Increase)



Financial Analysis

Inspection + Repair Net Profit	
$	11,543

Inspection + Repair Net profit	
$	28,187

Increased Profit After Recommended State	
$	16,644
(161% Increase)	

Wasted Wages	
$	2,655

Wasted Wages	
$	1,322

Decreased Wasted Wages	
$	(1,333)
(50% Decrease)	

THANKS!

Any questions?