Capstone Project Weekly Progress Report

Semester	Fall 2021
Course Code	AML 3406
Section	Section 2
Project Title	Restaurant Recommending Chatbot
Group Name	Group 3
Student names/Student IDs	Vignesh Kumar Murugananthan C0793760
Reporting Week	Week 12– 12/04/2021
Faculty Supervisor	Vahid Hadavi

1. **Tasks Outlined in Previous Weekly Progress Report**
   - Analyzed Business data and Review data
   - Explored the data to find various insights in them by exploratory data analysis
   - Grouped data based on the restaurant category and visualized them in order to understand how the data is represented.

2. **Progress Made in Reporting Week**
   - Firstly, we have split the train and test data sets with a composition of 80 and 20 percent respectively.
   - Then we have analyzed few algorithms with our train data.
   - The algorithms we used in our model building are:
     Logistic regression, Decision tree classifier, KNN, Random Forest, and Gradient boost.

```
#importing libraries of machine learning algorithm
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.naive_bayes import MultinomialNB

#importing libararies for selecting model
from sklearn.model_selection import cross_val_score
from sklearn.multiclass import OneVsRestClassifier
```

   - After modeling we have made the confusion matrix, which is one of the important metrics in classification algorithms.

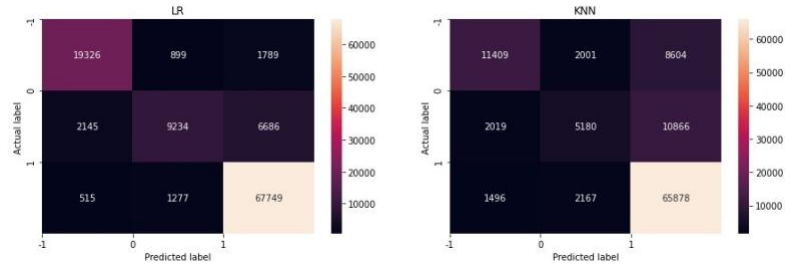
```
log = LogisticRegression(solver='liblinear',max_iter=10000)
knn = KNeighborsClassifier()
nb = MultinomialNB()
XGB = GradientBoostingClassifier()
dec_tree = DecisionTreeClassifier()
forest = RandomForestClassifier()

models = [('LR', log),
          ('KNN',knn),
          ('XGB',XGB),
          ('NB',nb),
          ('Decision Tree',dec_tree),
          ('Random Forest',forest)]
```

- With all the above given algorithms, we also used randomized search CV with Logistic regression to get the best parameters. With those parameters we have achieved accurate scores.

```
# Train all models
for name, model in models:
    now = datetime.now()
    print(f'{name} training started at {now.strftime("%H:%M:%S")}')
    model.fit(train_data,train_label)
    now = datetime.now()
    print(f'{name} training completed at {now.strftime("%H:%M:%S")}')
```

```
LR training started at 17:06:56
LR training completed at 17:10:40
KNN training started at 17:10:40
KNN training completed at 17:10:40
XGB training started at 17:10:40
XGB training completed at 17:15:37
NB training started at 17:15:37
NB training completed at 17:15:37
Decision Tree training started at 17:15:37
Decision Tree training completed at 17:20:05
Random Forest training started at 17:20:05
Random Forest training completed at 17:30:33
```

```
# Import confusion matrix
from sklearn.metrics import confusion_matrix

plt.figure(figsize=(15,15))
for i, model in enumerate(models):
    plt.subplot(3,2,i+1)
    y_predict = model[1].predict(train_data)
    cmatrix = confusion_matrix(train_label,y_predict)
    class_names=['-1','0','1'] # name  of classes
    # create heatmap
    sns.heatmap(pd.DataFrame(cmatrix, columns=class_names), annot=True, fmt='g')
    plt.title(model[0])
    plt.xticks(range(3),['-1','0','1'])
    plt.yticks(range(3),['-1','0','1'])
    plt.ylabel('Actual label')
    plt.xlabel('Predicted label')
plt.show()
```



	Name	F1 Mean	F1 STD	Accuracy Mean	Accuracy STD
0	LR	0.782199	0.002640	0.799316	0.002195
3	NB	0.772098	0.003115	0.774238	0.003062
2	XGB	0.711712	0.003057	0.751095	0.002098
5	Random Forest	0.679467	0.002069	0.748741	0.001590
4	Decision Tree	0.669946	0.004161	0.674849	0.004558
1	KNN	0.613156	0.003652	0.657535	0.003702

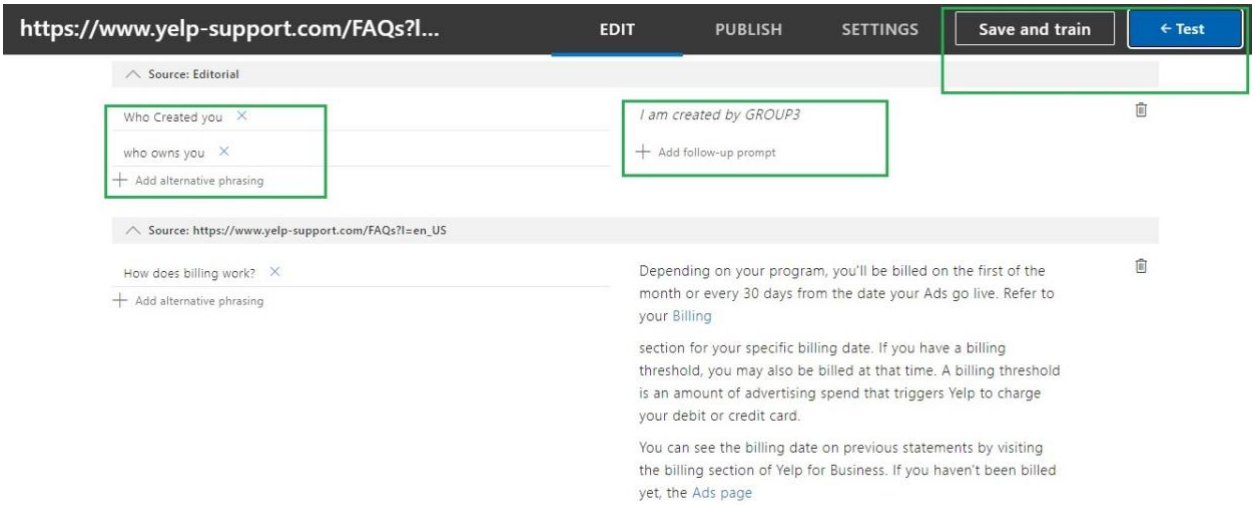
```
# Import RandomnsearchCV
from sklearn.model_selection import RandomizedSearchCV
from sklearn.linear_model import LogisticRegression

lr_classifier = LogisticRegression(max_iter=1000)

param_distributions = {'solver': ['newton-cg','lbfgs','liblinear','sag','saga'],
              'C': [100, 10, 1.0, 0.1, 0.01]}

random_search = RandomizedSearchCV(lr_classifier,
                        param_distributions=param_distributions,
                        scoring='f1_weighted',
                        cv=10,
                        verbose=10,
                        n_jobs=-1)

random_search.fit(train_data, train_label)
lr_opt = random_search.best_estimator_
```

```
[CV 1/10; 10/10] END .................C=1.0, solver=newton-cg; total time= 3.1min
[CV 2/10; 10/10] START C=1.0, solver=newton-cg................................
[CV 2/10; 10/10] END .................C=1.0, solver=newton-cg; total time= 3.0min
[CV 3/10; 10/10] START C=1.0, solver=newton-cg................................
[CV 3/10; 10/10] END .................C=1.0, solver=newton-cg; total time= 3.1min
[CV 4/10; 10/10] START C=1.0, solver=newton-cg................................
[CV 4/10; 10/10] END .................C=1.0, solver=newton-cg; total time= 3.2min
[CV 5/10; 10/10] START C=1.0, solver=newton-cg................................
[CV 5/10; 10/10] END .................C=1.0, solver=newton-cg; total time= 3.3min
[CV 6/10; 10/10] START C=1.0, solver=newton-cg................................
[CV 6/10; 10/10] END .................C=1.0, solver=newton-cg; total time= 3.0min
[CV 7/10; 10/10] START C=1.0, solver=newton-cg................................
[CV 7/10; 10/10] END .................C=1.0, solver=newton-cg; total time= 3.0min
[CV 8/10; 10/10] START C=1.0, solver=newton-cg................................
[CV 8/10; 10/10] END .................C=1.0, solver=newton-cg; total time= 3.1min
[CV 9/10; 10/10] START C=1.0, solver=newton-cg................................
[CV 9/10; 10/10] END .................C=1.0, solver=newton-cg; total time= 3.4min
[CV 10/10; 10/10] START C=1.0, solver=newton-cg...............................
[CV 10/10; 10/10] END ................C=1.0, solver=newton-cg; total time= 3.3min
```

```
print('='*50)
print("best params: " + str(random_search.best_params_))
print('best score:', random_search.best_score_)
print('='*50)
```

```
==================================================
best params: {'solver': 'newton-cg', 'C': 0.1}
best score: 0.7886700289213612
==================================================
```

- We have developed a chatbot using micro soft azure cognitive bot service.




```
foodon_city = input("Enter name of the city: ")
foodon_city = foodon_city.lower()
```

```
Enter name of the city:
```

Input the name of the city

```
foodon_city = input("Enter name of the city: ")
foodon_city = foodon_city.lower()
```

```
Enter name of the city: toronto
```

Input the name of Cuisine

```
cuisinelist=['indian','mexican','thai','japanese','italian','chinese','hungarian','german','vietnamese']
cuisinelist
```

```
7]: ['indian',
     'mexican',
     'thai',
     'japanese',
     'italian',
     'chinese',
     'hungarian',
     'german',
     'vietnamese']
```

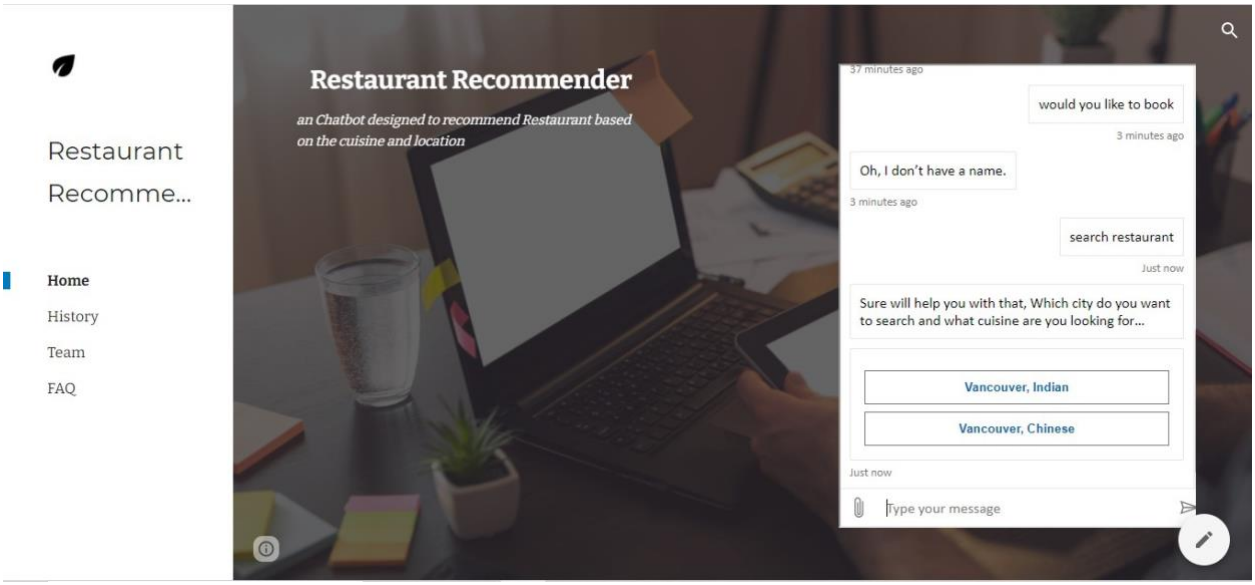
```
cuisine=input("Enter Cuisine: ")
cuisine = cuisine.lower()
```

```
Enter Cuisine:
```

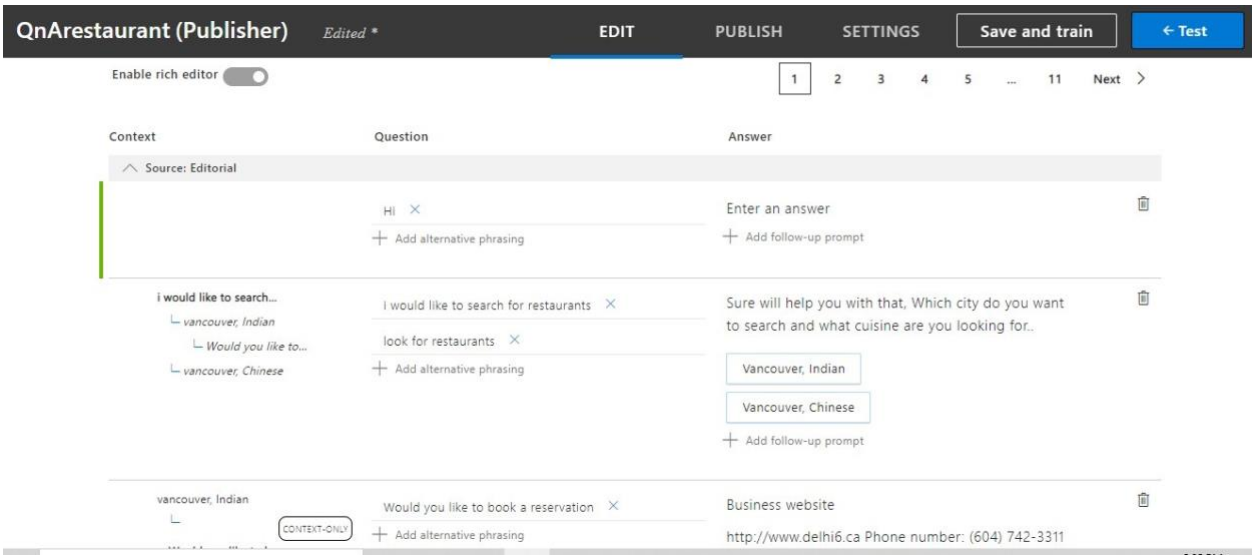
```
f1=final.sort_values('spm', axis=0, ascending=False).head(10)
f1
```

	name	address	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday	RestaurantsPriceRange2	spm
4658	dee	2013 yonge st	11:30-22:0	11:30-22:0	11:30-22:0	11:30-22:0	11:30-22:0	12:0-22:0	12:0-22:0	NaN	0.414553
2478	basil box	105 the pond road, unit r30	11:0-22:0	11:0-22:0	11:0-22:0	11:0-22:0	11:0-22:0	12:0-21:0	12:0-21:0	NaN	0.371071
2081	river tai restaurant	92 harbord street	NaN	NaN	NaN	NaN	NaN	NaN	NaN	$	0.346048
4762	bach yen	738 gerrard street e	12:0-21:30	12:0-21:30	12:0-21:30	12:0-21:30	12:0-21:30	12:0-21:30	12:0-21:30	$	0.337062
2722	silk restaurant & bar	446 parliament street	11:0-22:0	11:0-22:0	11:0-22:0	11:0-22:0	11:0-22:0	11:0-22:0	11:0-22:0	$$	0.333452
2685	thai mango	641 dupont street	11:0-23:0	11:0-23:0	11:0-23:0	11:0-23:0	11:0-23:0	11:0-23:0	11:0-23:0	NaN	0.329821
4155	saigon lotus	6 st andrew street	11:0-22:0	11:0-22:0	11:0-22:0	11:0-22:0	11:0-22:0	11:0-22:0	11:0-22:0	$$	0.323116
2359	sala modern thai kitchen & bar	1262 danforth avenue	11:0-22:0	16:30-22:0	16:30-22:0	16:30-22:0	16:30-22:30	12:0-22:30	12:0-22:0	$$	0.322934
3923	bolan thai cuisine	709 mount pleasant road	11:30-22:0	11:30-22:0	11:30-22:0	11:30-22:0	11:30-22:0	12:0-22:0	12:0-22:0	$$	0.311523
3499	kumo japanese restaurant	562 kipling avenue	11:0-22:0	11:0-22:0	11:0-22:0	11:0-22:0	11:0-23:0	11:0-23:0	12:0-22:0	$$	0.308854

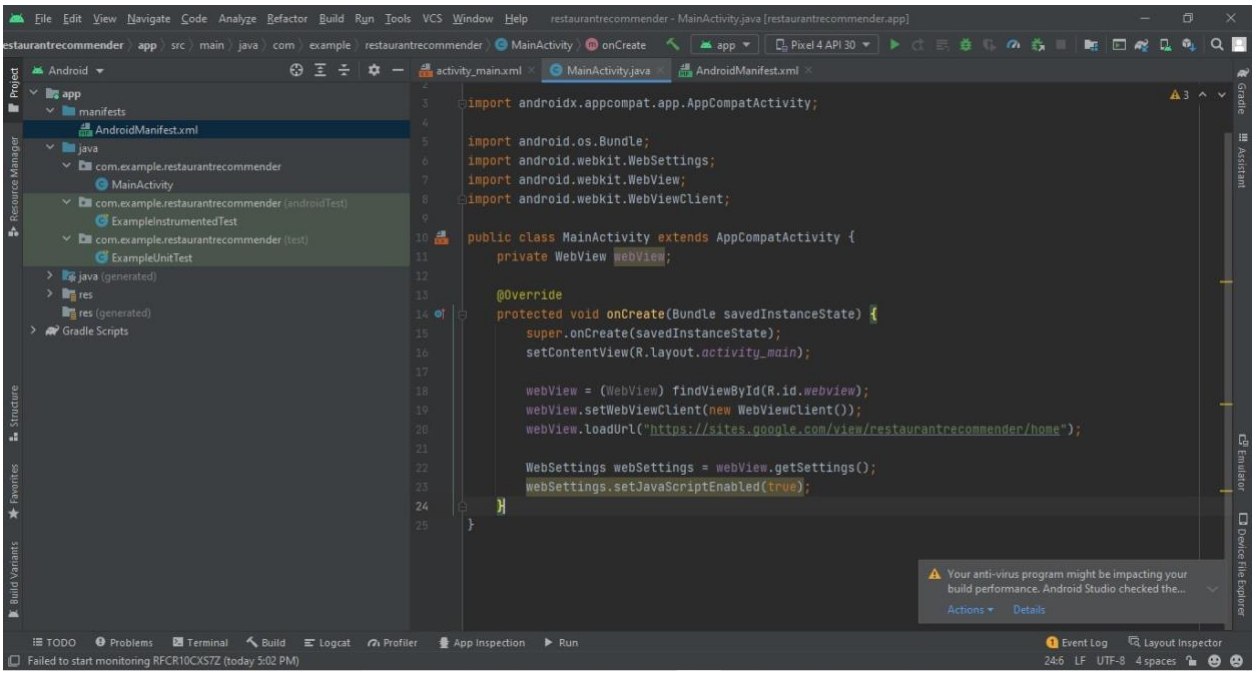
We created the chatbot using QnA maker with the help of knowledge base and for this we used the Chat Bot cognitive services from Azure.

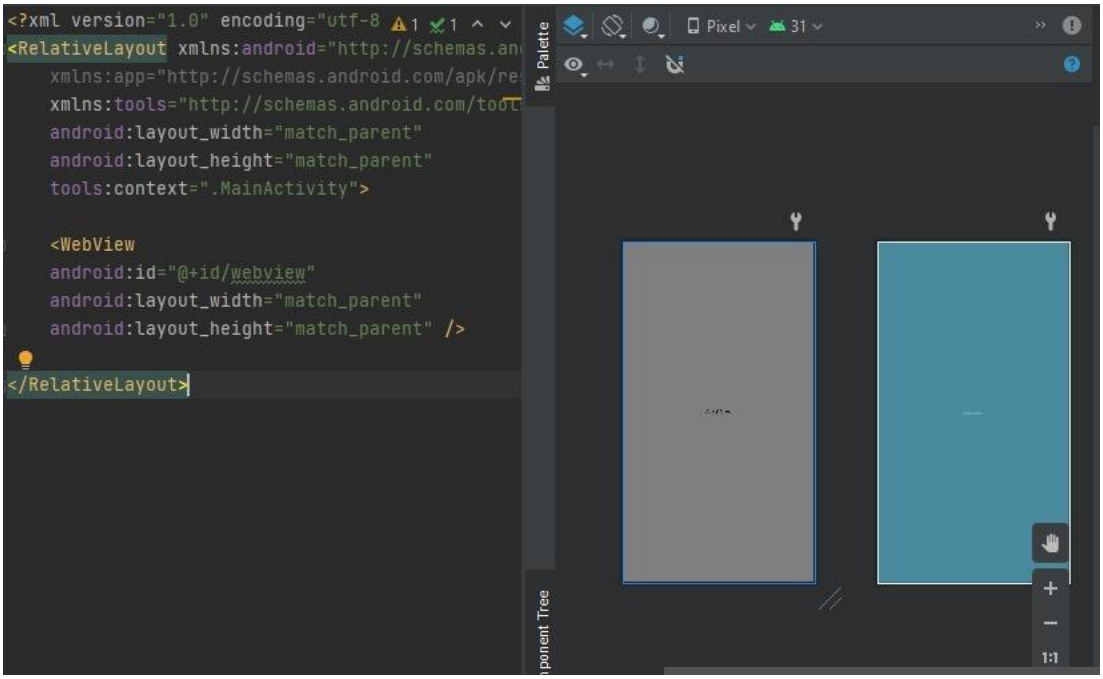


Name	Type	Last Viewed
restaurantQnA	QnA maker	3 hours ago
restaurantQnA	Resource group	3 hours ago
Azure subscription 1	Subscription	2 days ago



We are done with integrating the chatbot and creating a mobile app .
We have created android mobile app for our chatbot using java





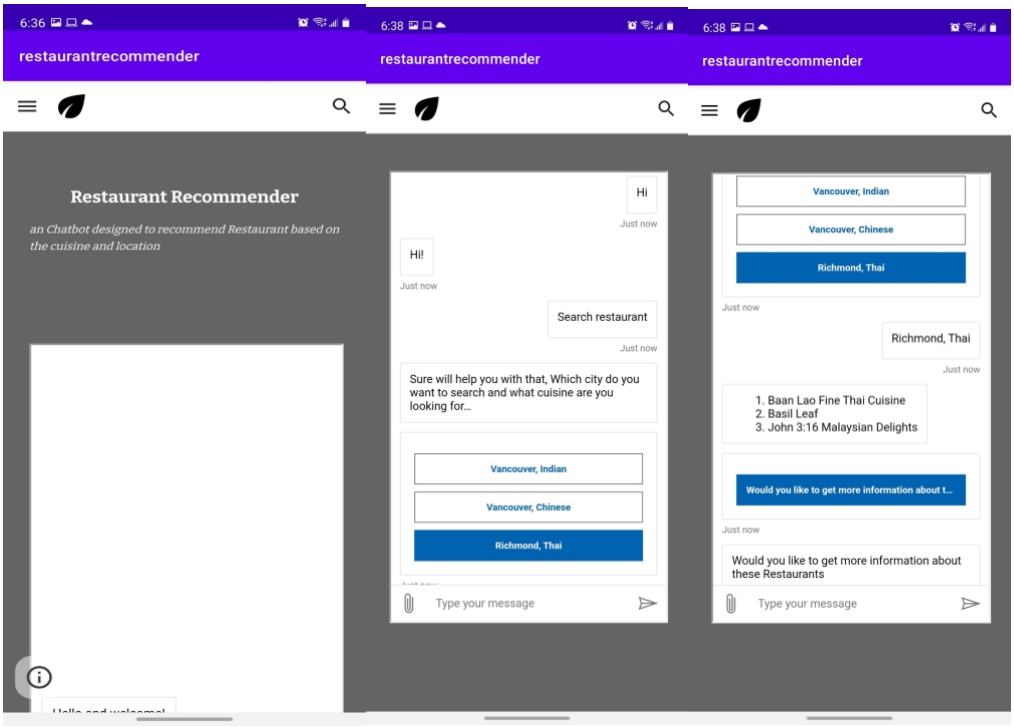
HAXM v7.7.0 Latest

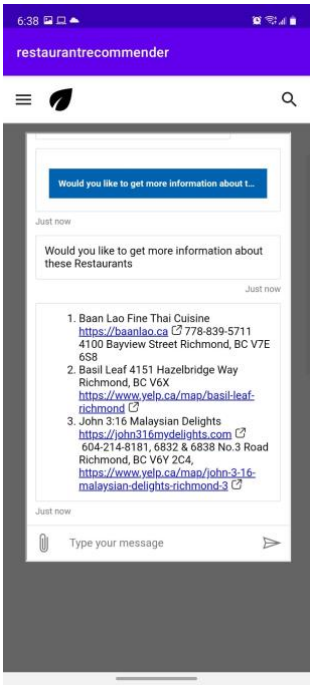
Change Log

- Added a new IOCTL to enable getting CPUID features for guest VCPUs (#383).
- Enabled all supported CPUID leaves to be configurable (#382).
- Enabled several features in CPUID emulation (#381).
- Migrated the CI service from Travis CI to GitHub Actions (#353).

Filename	SHA-256
haxm-windows_v7_7_0.zip	18dfc2edf3968acf20aea4cffdb2bddf7c448dca03e0b13ddf4fdf039e469a5a
haxm-macosx_v7_7_0.zip	d3fb74ca55e5312fc1c10b850c46689ac723572453c1bb3ed3f47680c7f504b7

Visualization of mobile app:





3. **Tasks to Be Completed.**
   - Integration Completed
   - Final Testing Done
   - Android app Built completed.
   - All Bugs resolved.
   - Improved Chatbot Responses.
   - Started with Reports and Slide creation.

4. Future improvements.
   - **Create a Ubiquitous chatbot application**
   - **Map Integration for Navigation**
   - **Faster results and implement Batch processing**
   - **Include more information about restaurants like amenities, open hours, etc.**

Overall Project Plan:

Restaurant Recommending Chatbot

TASK NAME	RESPONSIBLE	START	FINISH	DURATION (DAYS)	STATUS
Project Proposal					
Case study Analysis	Team	18-Sep	23-Sep	5	In Progress
Requirements Documentation	Vignesh/Murali	18-Sep	23-Sep	5	In Progress
Presentation Slides	Swathi	18-Sep	23-Sep	5	In Progress
Requirements Gathering					
SW Environment Setup	Team	23-Sep	1-Oct	8	Not Started
Data acquition - Scrapy, Twitter API	Vignesh/Varadha	23-Sep	1-Oct	8	Not Started
Requirement Analysis	Swathi	23-Sep	1-Oct	8	Not Started
Documentation	Swathi/Murali	23-Sep	1-Oct	8	Not Started
Development Phase I					
Model building	Vignesh/Murali	1-Oct	22-Oct	21	Not Started
NLP	Varadha	1-Oct	22-Oct	21	Not Started

STATUS
Complete
Overdue
In Progress
Not Started

Development Phase II					
Webapp Building	Swathi	23-Oct	12-Nov	20	Not Started
Buffer	Team	23-Oct	12-Nov	20	Not Started
Testing Phase I					
Unit Testing	Murali	13-Nov	19-Nov	6	Not Started
Integration Testing	Swathi	20-Nov	26-Nov	6	Not Started
Testing Phase II					
Functional Testing	varadha/murali	26-Nov	4-Dec	8	Not Started
Performance Testing	vignesh/swathi	5-Dec	13-Dec	8	Not Started
Final Presentation					
Report Generation/Documentation	Swathi/Murali	13-Dec	15-Dec	2	Not Started
Slides data collection	Varadha/vignesh	15-Dec	16-Dec	1	Not Started
Finalize Presentation	Team	16-Dec	17-Dec	1	Not Started
Approve Presentation	Team	17-Dec	18-Dec	1	Not Started

Individual Project plan:

Case study analysis

S/W environment setup

Development phase 1 – NLP

Development Phase 2 – Webapp Building

Testing Phase 2 – UNIT and Integration testing

HIGLIGHTED BELOW ARE MY RESPONSIBILITIES

Restaurant Recommending Chatbot

TASK NAME	RESPONSIBLE	START	FINISH	DURATION (DAYS)	STATUS
Project Proposal					
Case study Analysis	Team	18-Sep	23-Sep	5	In Progress
Requirements Documentation	Vignesh/Murali	18-Sep	23-Sep	5	In Progress
Presentation Slides	Swathi	18-Sep	23-Sep	5	In Progress
Requirements Gathering					
SW Environment Setup	Team	23-Sep	1-Oct	8	Not Started
Data acquition - Scrapy, Twitter API	Vignesh/Varadha	23-Sep	1-Oct	8	Not Started
Requirement Analysis	Swathi	23-Sep	1-Oct	8	Not Started
Documentation	Swathi/Murali	23-Sep	1-Oct	8	Not Started
Development Phase I					
Model building	Vignesh/Murali	1-Oct	22-Oct	21	Not Started
NLP	Varadha	1-Oct	22-Oct	21	Not Started

Development Phase II					
Webapp Building	Swathi	23-Oct	12-Nov	20	Not Started
Buffer	Team	23-Oct	12-Nov	20	Not Started
Testing Phase I					
Unit Testing	Murali	13-Nov	19-Nov	6	Not Started
Integration Testing	Swathi	20-Nov	26-Nov	6	Not Started
Testing Phase II					
Functional Testing	varadha/murali	26-Nov	4-Dec	8	Not Started
Performance Testing	vignesh/swathi	5-Dec	13-Dec	8	Not Started
Final Presentation					
Report Generation/Documentation	Swathi/Murali	13-Dec	15-Dec	2	Not Started
Slides data collection	Varadha/vignesh	15-Dec	16-Dec	1	Not Started
Finalize Presentation	Team	16-Dec	17-Dec	1	Not Started
Approve Presentation	Team	17-Dec	18-Dec	1	Not Started