Research Paper 3

HAYDEN VASS | DVP 2
E: HAVASS@STUDENT.FULLSAIL.EDU

MOBILE DEVELOPMENT
FULL SAIL UNIVERSITY

Application Used for Deconstruction

HOPPER

SECTION 1

The application being dissected in this research paper is an airline and traveling app called hopper. Hopper is an application that aims to streamline the travel process by providing its users with the cheapest flight and boarding arrangements. This application also provides the user with guidance by tracking the fluctuations in prices and notifies the user when to buy and when to hold out for a better price. This application was chosen for this research assignment due to its popularity. Hopper we developed in 2015, that same year they were named one of the best new apps by the New York times, Wall Street Journal, Tech Insider, TIME magazine and countless others. Hopper also won Apples Store Best of 2015, a Webby Award for best travel application and was feature on Google Plays bests apps of 2016. These accolades prove that Hopper has figured the equation to their success. This research paper will peel away the layers to one feature of this application, in an attempt to better understand what that equation holds.

FEATURE BEING DECONSTRUCTED

The feature being deconstructed is the ability to book a trip. This function allows the user to select their destination and provides them with data visuals of the best, most inexpensive days to travel. Once flights are booked Hopper does the same service but with places to stay. This function was chosen due to the its involvement with multiple data sources, and how important these relationships are. All these sources have to work in perfect unison, or the application simply will not function. If one piece of this process does not work at maximum efficiency then people have a very real potential of not only having their trip ruined, but losing a lot of time and money in the process.

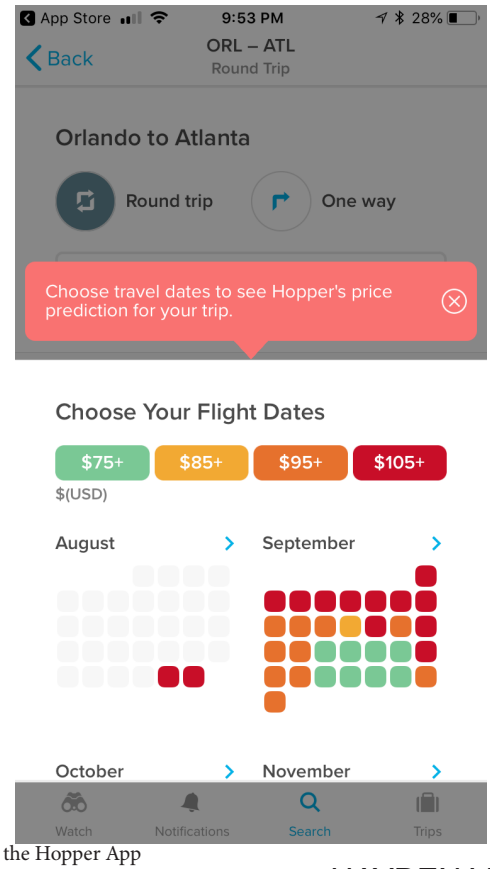

Image from the Hopper App

Data Sources and Application

SECTION 2

DATA SOURCES

1. Application user
2. Airlines
3. Hotels

HOW THEY ARE USED

Hopper utilizes multiple data sources to help its users achieve their in app goals. Three prominent sources being pulled are airline data, room and board data, as well as user input. From the consumer side, the primary data source is the user. When a user books a trip through the application they have to update their information with a first name, last name, birthday, gender and anything else an airline might need to know. This data then gets populated into a database and an account is created. Hopper also provides the user options of nearby airports. In this case the application is using geolocating the phone, which becomes the main data source, and cross referencing data from another database to show the user nearby airports.

Another data source is the airline companies. When a user sets their dates to travel they are setting query parameters. Hopper acts as a catalyst that takes these parameters, and through application programming interfaces (API), pulls back flight information from multiple airline databases. Hopper then performs a sort by price and returns the results to the user. While there is a primary data source that fills the airlines database with information, in this instance the airlines become the primary data source for Hopper.

Hopper also utilizes hotels as a primary data source in a similar fashion. The user sets their dates for travel and the application then turns that data into search parameters. But Hopper first cross references hotels in the area the user defined. This is most likely done through a database of hotels that participate in deals with Hopper. For example if a user wanted a hotel in Austin, Hopper would first run query of all their participating hotels in Austin. The app would then run a query against each hotels database for availability. This query would return the prices for the designated nights and Hopper would return those results to the user.

UML Tables and Possible Code

POSSIBLE UML CHARTS USED

SECTION 3

Provided below are some possible UML tables that Hopper might use to keep track of their users. A normalized table of payment methods exist to stream line optimization. The main users table contains all the common user information such as name and email. A septate table exist for payment info since a user can use multiple methods.

paymentInfo	
userID	varchar
paymentType	int
cardNumbe	int

users	
userID	varchar
firstName	varchar
lastName	varchar
phoneNumber	int
gender	char
age	int
email	varchar

paymentType	
Mastercard	int
Visa	int
Paypal	int
Discover	int

The tables below are some possible tables that could be found in a hotels data base. A guest table would track guest names and dates they are staying at the hotels and what room they would be in. A rooms table would keep track what rooms are available and what kind of room it would be. The main hotels table would hold key information pertaining to that hotel.

Rooms	
roomNumber	int
roomType	int
availability	int

Hotels	
hotelID	varchar
name	varchar
address	varchar
city	int
zipcode	char
phone	int
contact_poc	varchar

guest	
guestID	varchar
guestname	varchar
address	varchar
city	varchar
dates	date
room	int

POSSIBLE UML CHARTS USED

SECTION 3

Below are some possible UML tables an airline would use to track their flights. Various tables exist to track user payment methods and user information. A flights table exist to track flights and the remaining potential passenger slots on the flight. The ticket table is to track what kind of ticket the user as. Whether its a first class, business or economy. The airport table would also be important to an application like Hopper, so they can properly provide the correct flights for the correct airport.

paymentInfo	
userID	varchar
paymentType	int
cardNumbe	int

paymentType	
Mastercard	int
Visa	int
Paypal	int
Discover	int

ticketType	
economy	int
business	int
first	int

Passengers	
passengerID	varchar
firstName	varchar
lastName	varchar
phoneNumber	int
gender	char
age	int
email	varchar

ticket	
cost	int
type	int
flight	int

flights	
flightID	varchar
flightNumber	int
departDate	date
departTime	time
ariveDate	date
ariveTime	time
availability	int

airports	
name	varchar
city	varchar
state	varchar
zip	int

POSSIBLE METHODS AND CLASSES

SECTION 3

Hopper most likely uses a class for each of its data sources. A user class would be used to create an object for each passenger on a flight, or guest at a hotel. A user object would could contain information pertaining to that particular user. A users name, payment method, and potentially any other data that can be pulled from the Hopper database.

Class structure for a user

```
class User

public string UserID {get;set;}
public string FirstName {get;set;}
public string LastName {get;set;}
public string PaymentInfo {get;set;}
public user(_userID, _firstName,
_lastName, _paymentInfo)
{
UserId = _userId;
FirstName=_firstName;
LastName=_lastName;
PaymentInfo = _paymentInfo;
}
```

A flight class could exist that would contain an list of users or customers. The flight could inherit from an airline class. The flight object its self would contain details of that flight such as travel time, flight number, flight path, airports, and arrival and departure times. A method might exist to let the user know if spots are filling up, or i if availability is limited. A method could also be ran that compares the users price to the price of upgrading. If the price was in a certain range, the code could prompt the user that they could upgrade for what ever the difference was.

Class structure for a flight/ Method for availability

```
class Flight: Airline

public string FlightID {get;set;}
public DateTime DepartureDate {get;set;}
public DateTime DepartureTime {get;set;}
public DateTime ArrivalDate {get;set;}
public DateTime ArriavalTime {get;set;}
public int Availability{get;set;}

public flight (_flightID, _departureTime,
_departureDate, _arrivalTime, _arrivalDate,
_availability) : base (_airline, airport)
{
// fill object
}

public void FillingUp()
{
if(availability < 10)
{
cw(Spots filling up fast.)
}
}
```

POSSIBLE METHODS AND CLASSES

SECTION 3

A hotel class could be created as abstract and a room class could inherit from that. Each room would could have a type, a number, or section of the hotel (people will pay more with a view). Each room would have a customer attached with it. The customer object would contain user information such as guestID, name, and payment method.

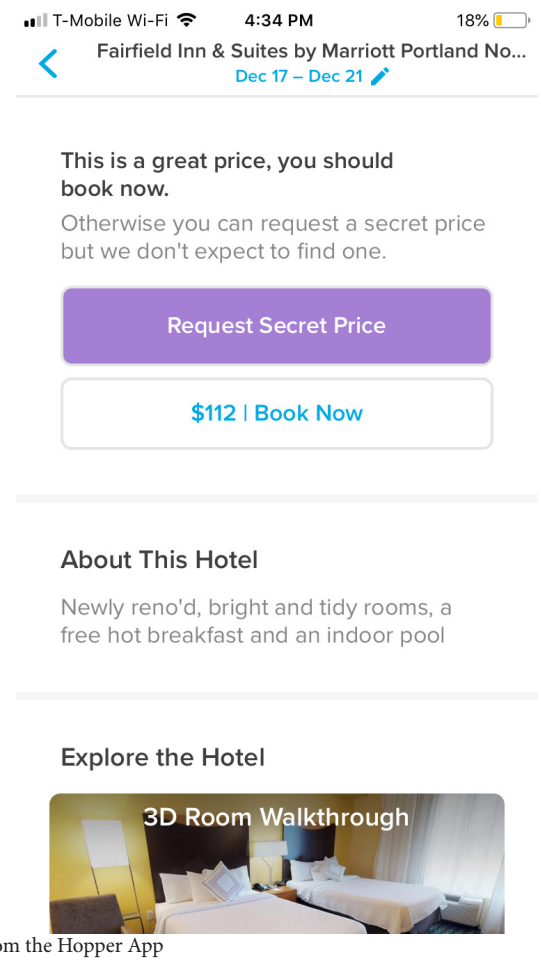

Image from the Hopper App

Class structure for a hotel room/ method for upgrading

```
class Room: Hotel

public string Guest {get;set;}
public DateTime CheckInDate {get;set;}
public DateTime CheckInTime {get;set;}
public DateTime CheckOutDate {get;set;}
public DateTime CheckOuttime {get;set;}
public int RoomType{get;set;}

public Room (// fill parameters): base (hotel)
{
// fill object
}

public void UpGrade()
{
if (upgrade - userCost < 150)
{
cw(Did you know you could upgrade for x amount more?)
}
```

REFERENCES

About. (2018). Hopper. Retrieved 9 September 2018, from https://www.hopper.com/corp/about.html

What is a Data Source? - Definition from Techopedia. (2018). Techopedia.com. Retrieved 9 September 2018, from https://www.techopedia.com/definition/30323/data-source

Kilroy, J. (2018). 100 of the Best Free Data Sources For Your Next Infographic. Column Five. Retrieved 9 September 2018, from https://www.columnfivemedia.com/100-best-free-data-sources-infographic

Understanding Data Sources. (2018). Docs.oracle.com. Retrieved 9 September 2018, from https://docs.oracle.com/cd/E17984_01/doc.898/e14695/undrstnd_datasources.htm

Data Source Types | Logi Analytics BI Encyclopedia. (2018). Logi Analytics. Retrieved 9 September 2018, from https://www.logianalytics.com/resources/bi-encyclopedia/data-source-types/

Primary Data Source - SAGE Research Methods. (2018). Methods.sagepub.com. Retrieved 9 September 2018, from http://methods.sagepub.com/reference/encyc-of-research-design/n333.xml

IBM Knowledge Center. (2018). Ibm.com. Retrieved 9 September 2018, from https://www.ibm.com/support/knowledgecenter/en/SSEP7J_10.1.1/com.ibm.swg.ba.cognos.ug_mfdm.10.1.1.doc/c_mfdm_typ_data_src.html

Types of Data Structures in Computer Science and Their Applications. (2018). Techspirited. Retrieved 9 September 2018, from https://techspirited.com/types-of-data-structures-in-computer-science-their-applications

algds. (2018). Cs.lmu.edu. Retrieved 9 September 2018, from http://cs.lmu.edu/~ray/notes/algds/

When to Fly and Buy on Hopper. (2018). Hopper. Retrieved 9 September 2018, from https://www.hopper.com/

HAYDEN VASS | DVP 2
E: HAVASS@STUDENT.FULLSAIL.EDU

MOBILE DEVELOPMENT
FULL SAIL UNIVERSITY