

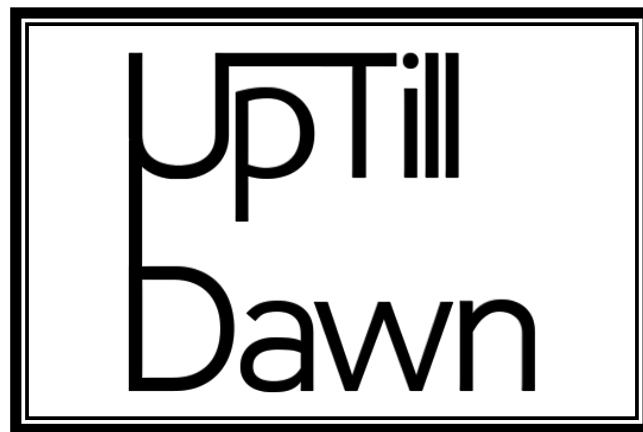


SINGAPORE UNIVERSITY OF
TECHNOLOGY AND DESIGN

Application Report







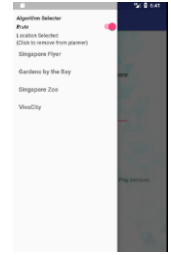
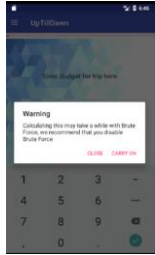
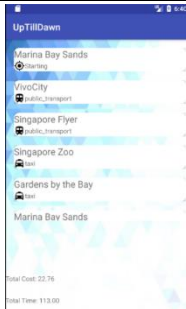
UpTillDawn

ISTD 50-001 FALL TERM 4



Travel Itinerary Mobile Application

	NRIC Name	Student ID	Cohort / Group
1	Chang Jun Qing	1002088	2/xx
2	Soong Cun Yuan	1002074	2/xx
3	Yang Lei	1002361	2/xx

	Features and Page design	Design
1	<ol style="list-style-type: none"> 1. Front page of the application presents the user with a selection menu uses an intent to proceed to the next screen, either to the attraction selection attractions page or the directions page. 2. The attractions button will bring you to the list of attractions available for the user to choose from. 3. The directions button will allow the users to find the best route to go to all the attractions selected 	 <p>Figure 1 Front Menu</p>
2	<ol style="list-style-type: none"> 1. In the list of attractions, we made use of a RecyclerView to display our attractions, and we allowed clicks and swipes on them. 2. On click, we start an Implicit Intent that shows us the location on Google Maps. 3. On a long press, it displays an image of the place. On swipe, the user can choose to add the item to their planner (swipe right) or to remove it (swipe left), animation is also added to provide visual cues when sliding the menu. 4. These data are stored in our SQLite Database so that they can be retrieved even when the user closes the app. 5. The “Show Planner” button displays an Alert Dialog that shows the current items added to the planner. The “Directions” button brings the user to the planner page that allows the user to put their budget for the trip. 	<div style="display: flex; justify-content: space-around;"> <div>  <p>Figure 2 Attractions page</p> </div> <div>  <p>Figure 4 Swipe Gesture</p> </div> </div> <div style="display: flex; justify-content: space-around;"> <div>  <p>Figure 3 On long press</p> </div> <div>  <p>Figure 5 Alert Dialog</p> </div> </div>
3	<ol style="list-style-type: none"> 1. On the planner page, the user can type in their budget for their trip. 2. If the user wishes to view what is in their planner so far, they can display it by pulling out the list on the DrawerView on the left. In the DrawerView, the user can choose to disable the Brute Force Algorithm as well (this is turned on by default to ensure a more accurate answer even though it takes a longer calculation time). 3. On tapping on the locations, the user can remove those locations from the planner. 4. When the “Travel Now “button is clicked, and AsyncTask is triggered to calculate the most optimal route base on our algorithm. Should there be more than 4 locations specified, the app will warn the user if he or she still has brute force algorithm turned on. While the task is running, a Progress Bar above will show an indeterminate progress bar that runs while the task is still calculating. 	<div style="display: flex; justify-content: space-around;"> <div>  <p>Figure 3.1 Planner Page</p> </div> <div>  <p>Figure 3.2 On side drawer (Brute Force switch and tap delete</p> </div> </div> <div>  <p>Figure 3.3 brute force warning</p> </div>
4	<ul style="list-style-type: none"> • Our final page displays the optimal route in a CardView which allows us to display the mode of transport in an image form. Allow a quick glance at the journey. 	 <p>Figure 4 CardView showing the route and transport as well as total time and money</p>

Algorithm Report

The complexity for Brute Force Algorithm: $O(n! \times 3^n)$

The permutation of n attractions gives us $n!$ orders. Each is further split into 3 mode of transport which gives us 3^n paths, which then gives us the complexity $O(n! \times 3^n)$

The complexity for Fast Approximate Solver: $O(n^2)$

We used a HashMap to retrieve the key-value pair which gives us an approximate complexity of $O(1)$. The algorithm solves for the answer by finding the nearest attraction to the current location and recursively call it until the order of the whole planner is determined which gives us the complexity of $O(n^2)$.

We then set aside 75% of the budget for taking a taxi (as taking the taxi is the most expensive but the fastest), we first allocate this 75% to the longest journey (which is the journey back to the hotel), and continuously check if the remainder is sufficient to cover the other distances. If insufficient, switch over to bus, and then to foot. The complexity of this is $O(n)$.

Thus, the final complexity for Fast Solver is $O(n^2)$

Our Algorithm should ideally find a reasonably good solution as taxi cost are usually about 3 or more times larger than the cost of public transport. By allocating the remaining 25% to public transport, we ensure that even in the case that taxi is used, not all the budget will be used on it such that the user has to walk on foot for the rest of the journey.

Comparing the 2 algorithms, with the following attractions: *[Marina Bay Sands, Singapore Flyer, Orchard Road, Singapore Zoo, VivoCity, Resorts World Sentosa, Marina Bay Sands]*

For Brute Force we get:

Order: *[Marina Bay Sands, Singapore Flyer, Orchard Road, Singapore Zoo, VivoCity, Resorts World Sentosa, Marina Bay Sands]*

Transport: *[Starting, Taxi, Foot, Public Transport, Taxi, Foot, Public Transport]*

Total time / Total travel fee: *[211.0, 19.99]*

For our algorithm we get

Order: *[Marina Bay Sands, Resorts World Sentosa, Singapore Flyer, VivoCity, Orchard Road, Singapore Zoo, Marina Bay Sands]*

Transport: *[Starting, Taxi, Public Transport, Public Transport, Public Transport, Public Transport, Public Transport]*

Total time / Total travel fee: *[289.0, 15.93]*

The difference in timing is 78 minutes and the cost are \$4.06

Brute Force provides us with the most optimal solution, however, when large amount of attractions is provided, the time required to calculate them is highly inefficient. Our own algorithm on the other hand, is not as accurate as the Brute Force Method, however, it generates the result in a significantly quicker time.

Features incorporated

1. AsyncTask (Background algorithm utilization)
2. AlertDialog
3. RecyclerView with gesture support
4. NavigationView with RecyclerView
5. Singleton Class
6. SQLite Database
7. CardView to display result
8. Progress bar to show calculation
9. Animated Selection

END