

**CMPT 363**  
**Project Part 1**

**Name** - Barundeeep Gambhir

**SFU email** – bsg10@sfu.ca

**SFU ID** - 301437428

Contents:

Part 1a) Heuristics Evaluation + Design Requirements Specification

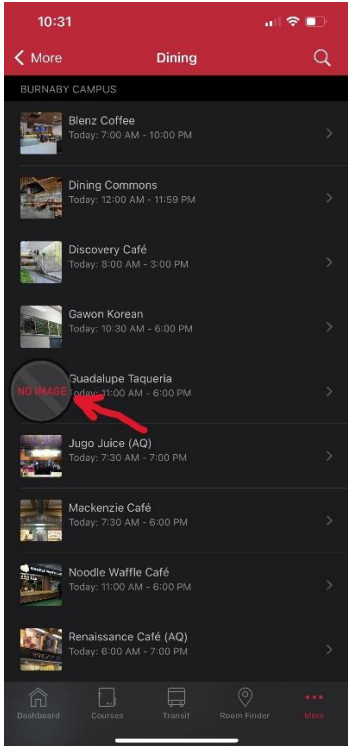
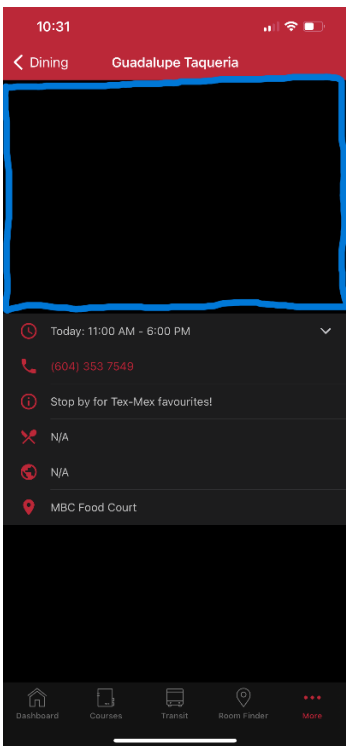
- ➔ Heuristics Evaluation
- ➔ Summary
- ➔ Context Identification
- ➔ User Identification
- ➔ Functional Requirements
- ➔ Non-Functional Requirements

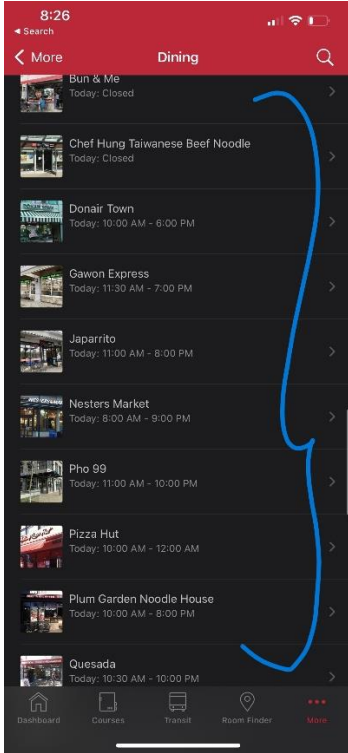
Part 1b) Low-Fidelity Prototypes

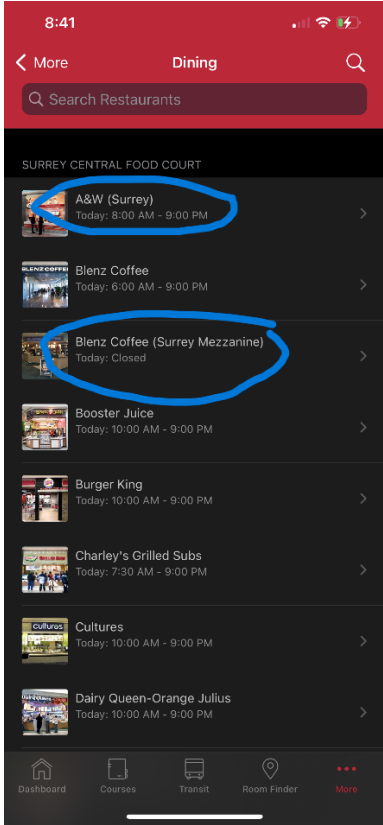
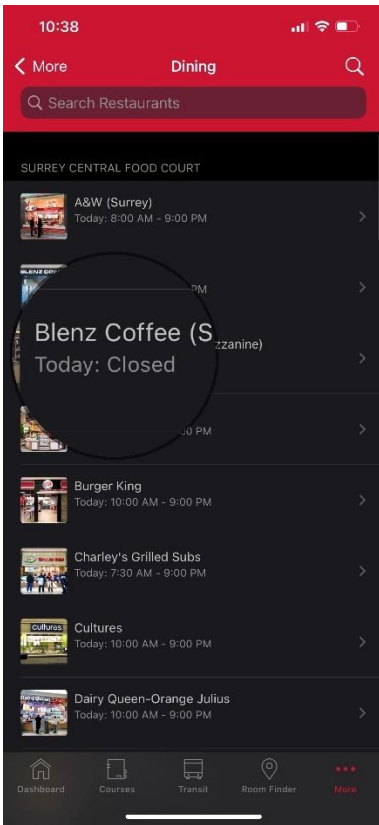
Appendix

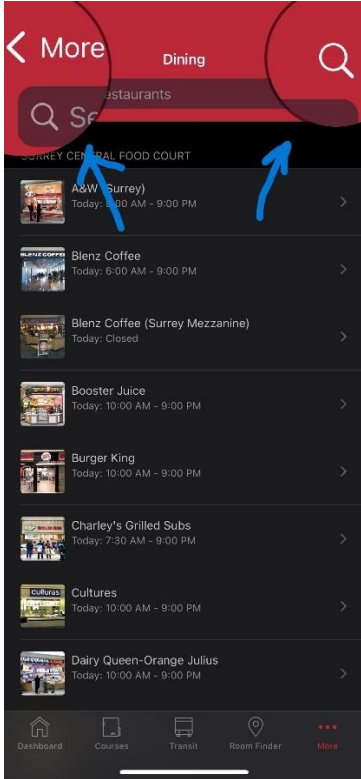
## Part 1a) Heuristics Evaluation + Design Requirements Specification

### Heuristics Evaluation

#: 1	<b>Problem/Good:</b> Problem
<b>Name:</b> No Image Criteria	
<b>Relevant heuristic:</b> Consistency and Standards	
<b>Evidence of issue:</b> <div data-bbox="203 451 548 1192"></div> <div data-bbox="581 451 924 1192"></div>	
<p><b>Detailed explanation:</b> Upon opening the dining feature (More-&gt;Dining), we find ourselves with a list of restaurants separated by campus. Each restaurant is either associated with an image on its left or with a No Image icon in case there isn't any image provided for the restaurant (as seen in Image 1.1). However, such is not the case once we click on the restaurant (in this case – Guadalupe taqueria) having no image. The restaurant info looks like it is floating which might confuse the users.</p>	
<p><b>Severity or Benefit (minor, major, critical):</b> severity of level 2. (Minor) <b>Justification:</b> The given issue seems to confuse its users who may think that some information is kept hidden from them but might not understand exactly what. Compared to Google Maps which shows a blurry grey image in case there is no picture of a particular restaurant, this feature doesn't maintain the standard consistency.</p>	
<p><b>Possible solutions and/or Trade-offs:</b> Solution: Providing a No Image picture on top would keep the aesthetics of the restaurant info maintained. Trade-off: None, we are only reusing the initial No Image Icon (in Image 1.1) as a full picture in the blue border area (Image 1.2).</p>	

#: 2	Problem/Good: Problem
Name: Long list of Restaurants	
Relevant heuristic: Flexibility and Efficiency of Use	
<p><b>Evidence of issue:</b></p>  <p>Image 2.1</p>	
<p><b>Detailed explanation:</b> Upon entering the dining feature, users must scroll down to the bottom (worst case) of the list to view the restaurant of their choice at their current/ preferred location.</p>	
<p><b>Severity or Benefit (minor, major, critical):</b> Severity of level 3. (Major)</p> <p><b>Justification:</b> As seen in Image 2.1, there is a list of more than 50 restaurants inside the dining feature among a total of four campuses. While browsing for restaurants at a particular campus, a user must scroll down the list and manually find the preferred campus location.</p>	
<p><b>Possible solutions and/or Trade-offs:</b></p> <p>Solution: use the filter option to allow users to choose the campus of their preference</p> <p>Trade-off: addition of a new version within an existing feature, would lead to a new version release of the application</p>	

#: 3	Problem/Good: Good
Name: Restaurant opening hours	
Relevant heuristic: Visibility of System Status	
Evidence of issue:	
	
Image 3.1	Image 3.2
<p><b>Detailed explanation:</b> On browsing the dining feature inside the SFU Snap App, we find that the information below the restaurant name shows whether that restaurant is closed on the current day (Image 3.2). In case a restaurant is not closed, its opening hours are mentioned (as shown in Image 3.1)</p>	
<p><b>Severity or Benefit (minor, major, critical):</b> Benefit of level 3 (Major).</p> <p><b>Justification:</b> This proves to be a helpful feature for the users as its dining feature provides appropriate feedback timely, and interacts with the user in real-time, potentially providing the status of restaurants.</p>	
<p><b>Possible solutions and/or Trade-offs:</b></p> <p>Improvement: In addition to the timings (in the case of opened) and closed today, estimated wait time at the restaurant can be added just below the “Today” line.</p>	

#: 4	Problem/Good: Good
Name: Marked back and search	
Relevant heuristic: User Control and Freedom	
<p><b>Evidence of issue:</b></p>  <p>Image 4.1</p>	
<p><b>Detailed explanation:</b> The Dining feature shows a way to go back to more options, and another button to search for a particular restaurant.</p>	
<p><b>Severity or Benefit (minor, major, critical):</b> Benefit of level 3. (Major)</p> <p><b>Justification:</b> Users always need a clear way to go a page back and the back to more option on the top left (Image 4.1) notifies the users. Moreover, a search icon on the top right helps the user to search for a restaurant.</p>	
<p><b>Possible solutions and/or Trade-offs:</b></p> <p>Improvement: if we search for a campus, no information is present, searching for a campus should show the list of restaurants for that campus.</p> <p>Another improvement is to pop up the keyboard to search for the restaurants upon pressing the search icon.</p>	

## Summary

### What has been evaluated?

SFU Snap app's © Simon Fraser University – Version 2.14 (116), Dining feature has been evaluated as a Part 1a) deliverable of the project part 1. The feature can be of use to SFU students and faculty strictly. So, the only type of users this app targets is a part of SFU. After using the application on both iOS and Android devices, two usability problems where one of [Nielsen's ten heuristics](#) was violated in each example in the feature and two examples of good usability throughout the dining feature of the SFU Snap App have been mentioned. A detailed explanation, justification, and possible solution/ improvement along with Trade-offs are further provided.

### When did the evaluation take place?

The application's Dining feature was first used for roughly 12 days in a row, just after project part 1 was posted to carry out a detailed evaluation. After that, evidence was collected, and descriptive documentation followed.

### Where did the evaluation take place?

As mentioned by the course instructor to use the application's dining feature under different conditions and geographical locations, the evaluation took place in the SFU Surrey and Burnaby campuses along with certain locations in the Vancouver downtown area as well. Moreover, the app was used both in guest mode and after signing in to compare the differences.

### How has the dining feature been evaluated?

As mentioned above, different features of the app's dining feature were tested by me as the only user for this evaluation while considering the ten heuristics and making note of violations of any of these heuristics or when these were seen being used within the feature.

Steps of evaluation:

- Signed into the application with SFU credentials.
- Used the application's dining feature in both light and dark modes.

### Main Findings

- The fourth heuristic, Consistency and Standards, and the seventh heuristic, Flexibility, and Efficiency of Use were violated in the dining feature of the SFU Snap App.
- The first heuristic Visibility of System Status and the third heuristic, User Control and Freedom were seen in action in the dining feature of the SFU Snap App.

## Context Identification

**Who:** The SFU students and faculty can use the SFU snap app's dining feature and the new features within the dining that are being introduced ahead in the documentation.

**When:** The dining feature of the application is always accessible to the SFU students and faculty during the day. An Internet connection (Wi-Fi or Mobile data) is recommended to accurately view the status of a restaurant.

**What:** Some additional features are added to the dining feature that would allow the users to take full advantage of the feature.

**How:** By providing the users with a way to filter the restaurants concerning the campus of their preference to narrow down their search in finding the perfect restaurant. Also, getting a way to reach their restaurant of choice. Moreover, getting to know the estimated wait time at a particular restaurant would allow the users to save their precious time

**Why:** To improve the overall user experience for the dining feature.

## User Identification

**First Persona:** Jane Doe is a first-year international student at SFU pursuing a Bachelor of Science degree in Computing Science. Due to scheduling reasons, she ended up taking lectures at both the Surrey and Burnaby campus. She often finds it difficult to search for newer restaurants in her proximity as she likes to eat something new and different each day which would also be beneficial for her health.

Upon addition of the filter feature within the dining feature, she could easily switch between the two campuses and doesn't necessarily get confused between the variety of options available. For an international student, understanding the routes of a new country is difficult as well, hence the addition of the maps feature within the dining feature will allow Jane to get the fastest possible route to the restaurant of her choice

**Second Persona:** Tom Hanks is a Theatre student at SFU and loves to post food vlogs on his YouTube channel. Tom loves to roam about all three campuses of SFU and try different restaurants to get as much content for his YouTube channel as possible. Moreover, Tom also wants to know what the waiting time is at a particular restaurant so that he can spare himself the pain.

Similarly, the addition of a filter option will allow Tom to navigate easily through all three campuses, and the addition of a map feature will allow him to reach his destination easily. Moreover, adding the estimation of wait time at a particular restaurant will save him some time as well

#### Functional Requirements:

- 1) The dining feature interface allows the user to filter within the feature of the campus in which they want to browse the restaurants. (non-trivial and doesn't exist in the current version of SFU Snap App)
- 2) Inside the restaurant info, the feature must direct the user to the restaurant location by using Google Maps API

#### Non-Functional Requirements:

- 1) When the user selects the search icon on the top right of the dining feature, the keyboard opens which allows users to enter the name of the restaurant
- 2) The estimated wait time is visible for the restaurants in the dining feature.

#### Next Steps

A strict pattern of User Centered Design must be followed which involves

Analysis -> Design -> Evaluation -> Implementation

After specifying the context of the design requirements, specifying two personas of targeted users, and gathering both the functional and non-functional requirements the next steps include creating the prototype designs for our theoretical requirements. For the scope of this project, we will stick to creating only the low-fidelity prototypes for all our gathered requirements and try to incorporate all 4 requirements (both functional and non-functional) into our prototypes. Building multiple prototypes will allow us to have a better perspective on including all the requirements in a rather more succinct manner. In a real-world scenario, further steps include building medium/mid-fidelity prototypes (MFPs) and further High-Fidelity Prototypes (HFPs) as well which will be like the actual implementation but will differ in many ways (may be cheaper but not cheaper than LFPs and MFP's). Also, in case of any changes in the requirements throughout the process, we should reflect those changes as well along the way and include them in our prototype building as well. Potential discussions must take place in case of any changes in the requirements.

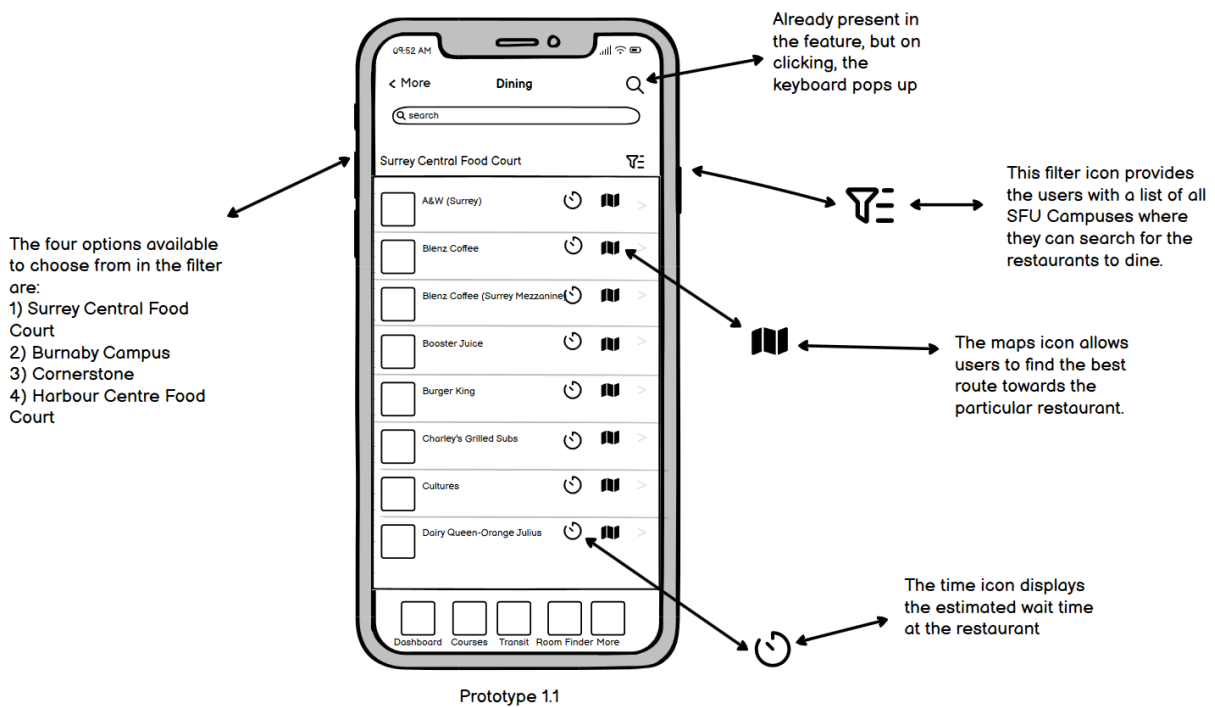


## Part 1b) Low-Fidelity Prototype

### Prototype 1

- ➔ Consists of all 4 requirements (2 functional and 2 non-functional)
- ➔ For functional requirements, i.e., the addition of a filter option to select the preferred campus, and a maps option to get the fastest route possible toward a particular restaurant is mentioned in the Prototype 1.1 Image screenshot. Further, the filter option's working is also shown in Prototype 1.2 Image on the next page. And the maps option is also shown in Prototype 1.3 Image
- ➔ The non-functional requirements, i.e., the addition of the estimated wait time of the restaurant and the opening of the keyboard upon clicking the already present search icon, is mentioned in the Prototype 1.2 image and Prototype 1.1 Image respectively.
- ➔ As mentioned in Balsamiq's prototype design, the functional requirement of adding the filter is mentioned by the placement of the filter icon on the top right just below the search bar.
- ➔ The functional requirement of the addition of maps to get the fastest possible route is added alongside the restaurant's tab on the main page of the dining feature and is mentioned by the map's icon.
- ➔ The non-functional requirement of opening the keyboard upon clicking the search icon on the top right is mentioned in Prototype 1.1 Image
- ➔ The non-functional requirement of the addition of estimated wait time at a given restaurant is mentioned as a time icon alongside the restaurant tab on the main page in prototype 1.1 Image

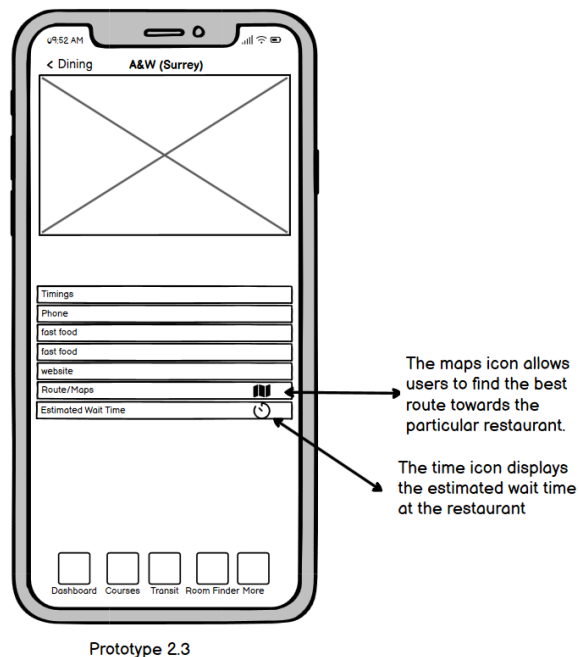
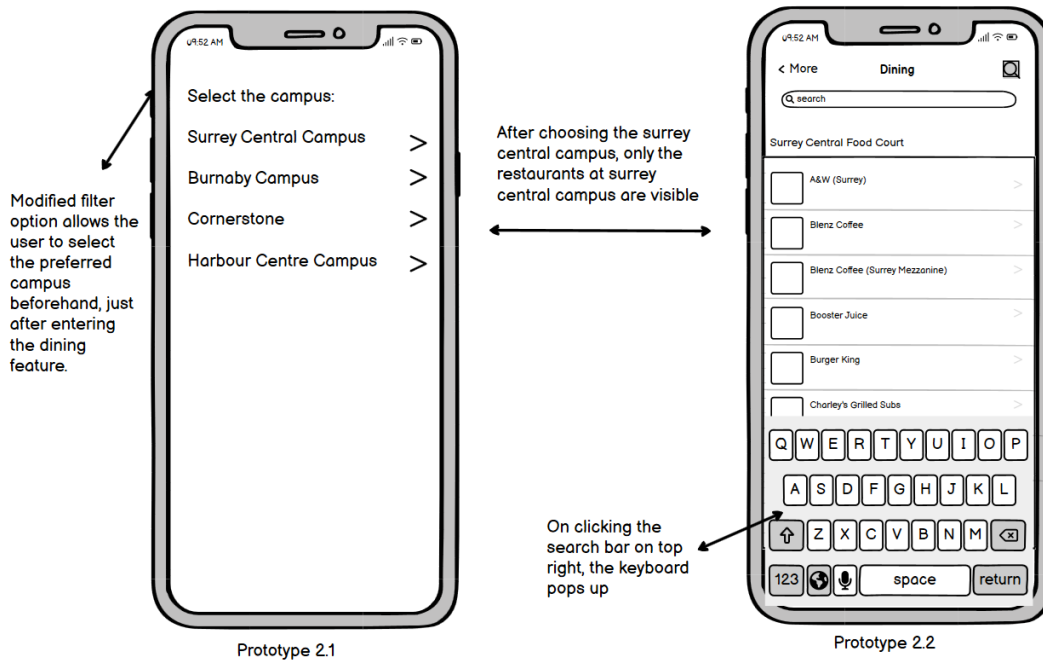
The screenshots are posted on the next page.



## Prototype 2

- ➔ Consists of all 4 requirements (2 functional and 2 non-functional)
- ➔ For functional requirements, i.e., the addition of a filter option to select the preferred campus, and a maps option to get the fastest route possible toward a particular restaurant is mentioned in the Prototype 2.1 Image screenshot and Prototype Image 2.3 respectively. Further, the filter option's working is also shown in Prototype 2.2 Image on the next page.
- ➔ The non-functional requirements, i.e., the addition of the estimated wait time of the restaurant and the opening of the keyboard upon clicking the already present search icon, is mentioned in the Prototype 2.2 image and Prototype 2.3 Image respectively.
- ➔ As mentioned in Balsamiq's prototype design, the functional requirement of adding the filter is mentioned by adding a new page that opens just after entering the dining feature, where the user is allowed to choose between the campuses to find their desired restaurant. (Prototype Image 2.1)
- ➔ The functional requirement of the addition of maps to get the fastest possible route is added on the restaurant page as shown in Prototype 2.3
- ➔ The non-functional requirement of opening the keyboard upon clicking the search icon on the top right is mentioned in Prototype 2.2 Image
- ➔ The non-functional requirement of the addition of estimated wait time at a given restaurant is mentioned as a time icon in a particular restaurant page as shown in prototype 2.3 Image

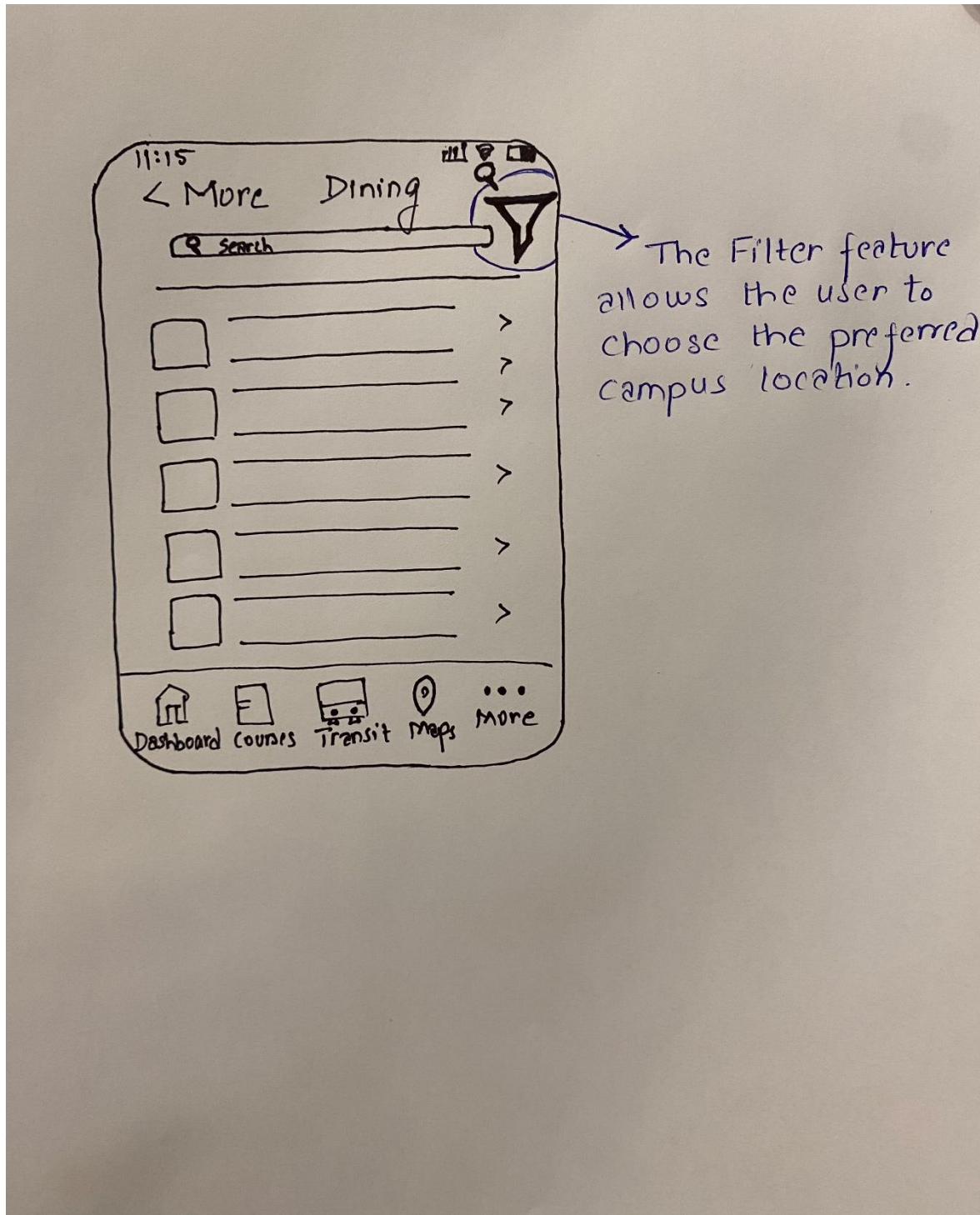
The screenshots are posted on the next page.



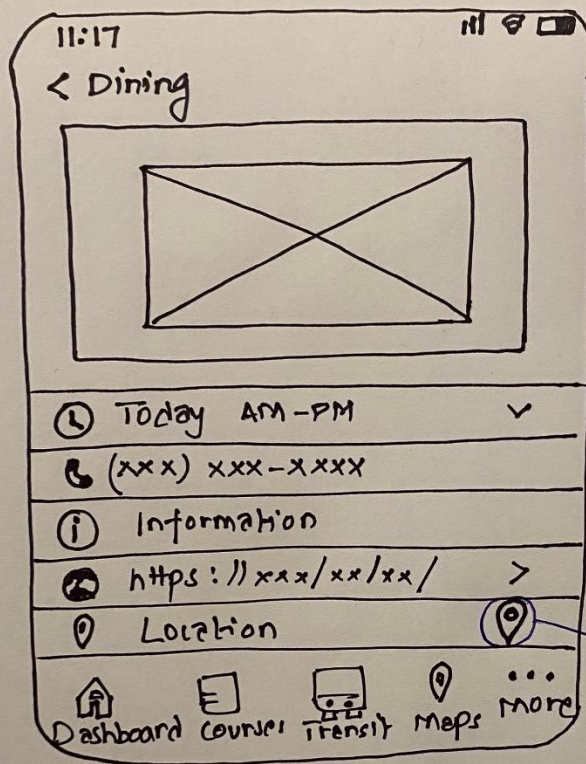
## Appendix

The following are the two sketches for the two functional requirements

### 1) Functional Requirement 1



## 2) Functional requirement 2



An additional map feature will allow users to get the shortest path towards their desired restaurant.