W1 PRACTICE

*MOBILE **USABILITY ANALYSIS***

In this practice, you will step into the shoes of a UX designer and critically evaluate a mobile application.

Usability and **design** are key factors that influence how users interact with an app, and understanding these aspects is essential for creating effective and enjoyable digital experiences.

Learning objectives

- ✓ Recognize inefficiencies or potential **usability issues** in **user flows**
- ✓ Evaluate a mobile application based on **Nielsen's 10 Usability Heuristics**
- ✓ Critically analyze the **app's design** strengths and weaknesses

How to submit?

- ✓ Attach your **report t**o the MS Team assignment and **turn it in**

Some references about UX/UI!

USABILITY THEOERY

Jacob 10 heuristics
Don't Make Me Think Book
Introduction To Usability
https://www.kickassux.com/ux-library/ux-process

CASE STUDY EXAMPLES

Kotak Case Study
ContractCar Case Study

STEP 1 –APP SELECTION & INSTALLATION

Select and install an existing mobile application.

WARNING: Choose an application related to 1 of following business types:

- **Transportation app** *(ex: Grab or BookMeBus)*
- **Food Delivery app** *(ex: Delishop, NHAM24)*

App Name	*Grab*
Business Type	Transportation app

STEP2 – TASK SELECTION

Identify a relevant task on the application.

WARNING: The task should be **complex enough** (take multiple steps to complete)

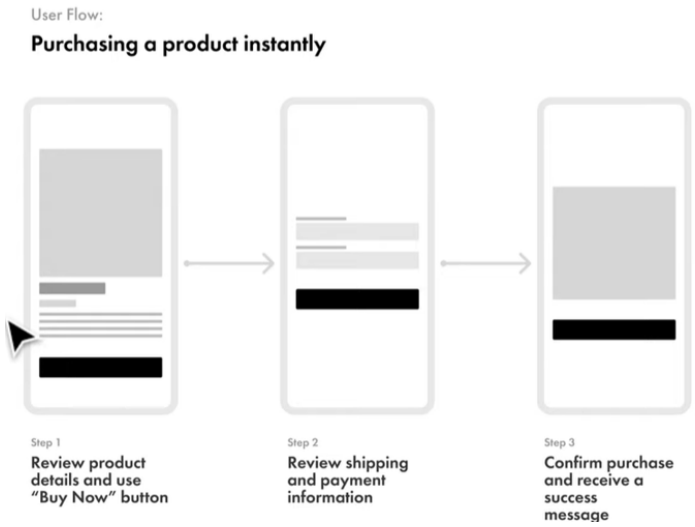
Task	Book a Tuk-Tuk from my place to CADT
Targeted user	Grab is for everyone, especially students, teachers, and staff.
Frequency	This task is done every day, sometimes many times a day, especially during busy hours when people need a ride or order food.
Importance	*Select the relevant option*: - **The task is essential for the primary purpose of the application** (Booking a ride it is a core function of the Grab app)

STEP 3 – USER FLOW TO PERFORM THE TASK

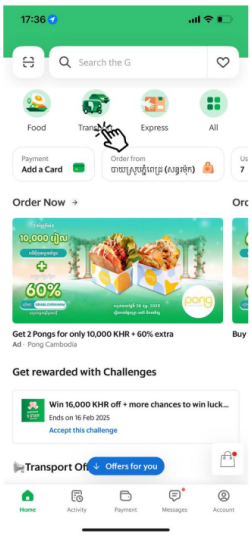
Write the user flow (sequence of low-definition wireframes) to complete the chosen task.

How to write a user flow?

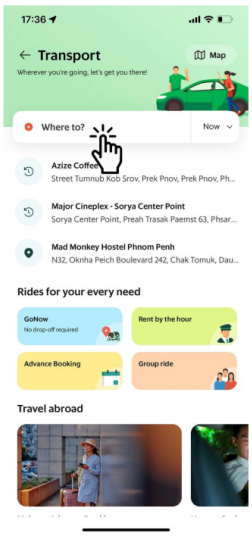
INSERT FIGMA LINK or SNAPSHOT HERE



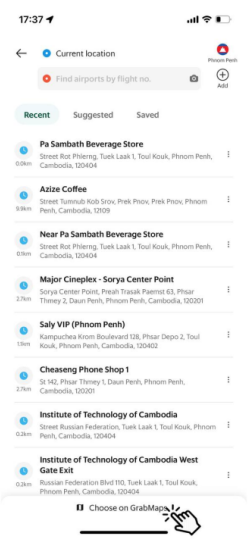
The user flow investigates the actions required by a user to complete the task.



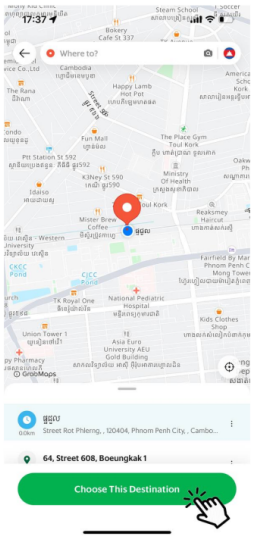
1. Choose the ride option



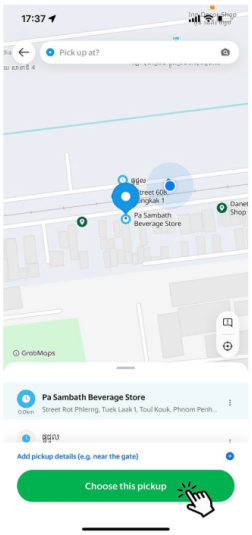
2. select location where to go



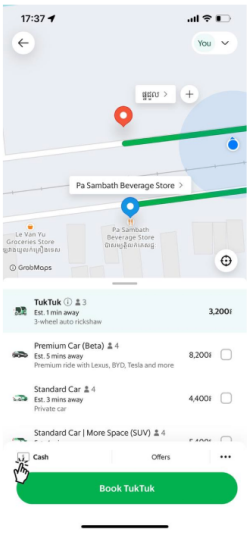
3. Choose location on the map



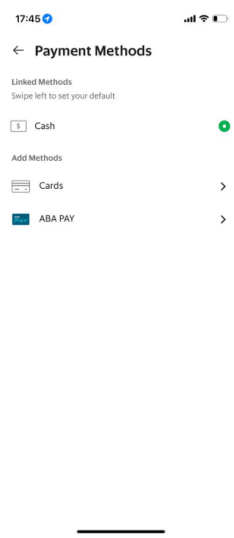
4. Choose your destination



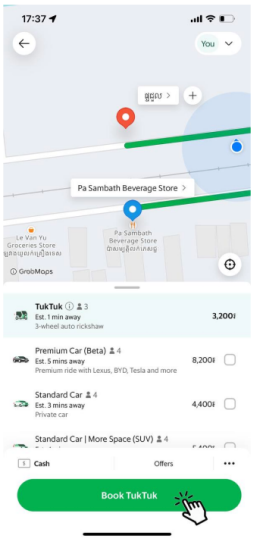
5. Choose location where to pick up



6. Select payment method



7. Choose your payment

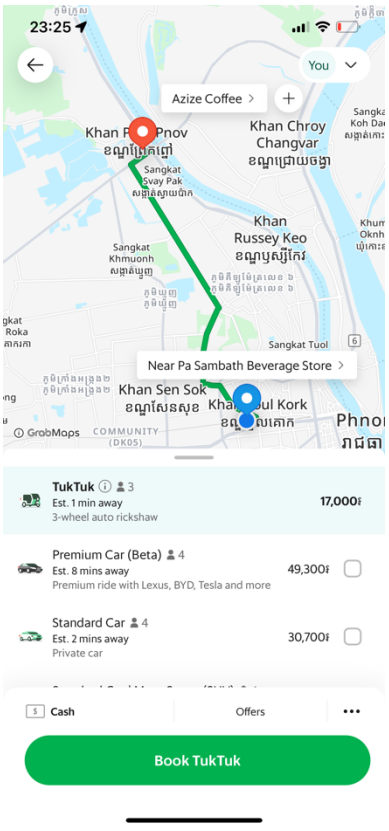


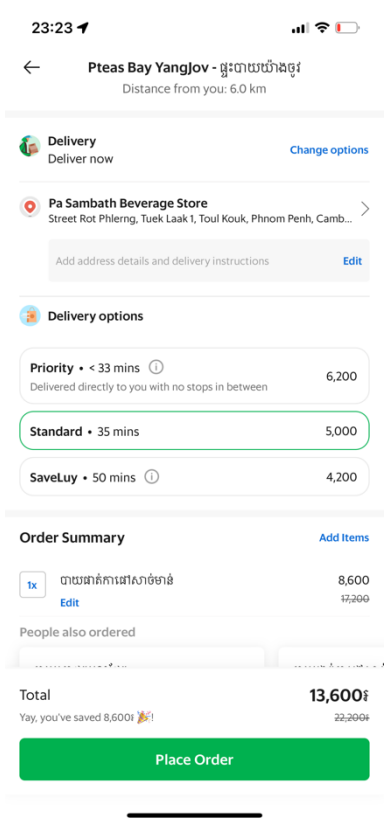
8. Booking Tuk Tuk

Analyze the user flow

- Is the **user flow intuitive** and easy to follow for a first-time user?
- Are there **unnecessary** steps or redundant actions in the process?
- Are there points in the flow where the user **might feel confused** or encounter difficulties?

- **Yes,** the Grab app is designed to be simple and clear, even for first-time users. The layout is straightforward, and users can easily find the options they need.
- For example, in the **payment process**, users sometimes have to confirm payment details multiple times (once when selecting the payment method and again when confirming the order).
- Using different button shapes, like one rectangle and one with rounded corners, makes the design look messy and may confuse users.
- If the app uses different terms for similar actions (e.g., "Booking Tuk Tuk" for rides and "Place order " for food), users may get confused about what these terms mean.





How **many steps** are needed to complete this task?

Do you think the **user flow** is relevant, regarding the **importance and the frequency** of this task?

- **For ride booking:**

1. Open the Grab app.
2. Choose the ride option (e.g., car, tuk-tuk).
3. Set the pick-up location (either by entering an address or selecting from the map).
4. Set the drop-off location (CADT or another destination).
5. Confirm the ride details (check pricing, time, etc.).
6. Select payment method (e.g., cash, ABA).
7. Book the ride (tap “Booking Tuk Tuk”).

- For Food Delivery:

1. Open the Grab app.
2. Select the restaurant or item you want to order.
3. Set the delivery address.
4. Add items to cart.
5. Confirm the order and payment method.
6. Place the order.

STEP 4 – USABILITY HEURISTICS

Apply recognized usability principles, such as Nielsen’s 10 Usability Heuristics, to identify potential issues.

To guide your analysis:

- See below the check list points for each heuristic

- You can also get inspirations on other case studies:
  - Kotak Case Study
  - ContractCar Case Study

WARNING: You need to identify at **least 3 relevant UX/UI issues** and refer to the right Heuristic.

Usability issue	Inconsistent Button Labels (e.g., "Booking Tuk Tuk" vs. "Place Order").
Related Heuristic	Consistency and Standards
Proposed remedial (optional)	

Usability issue	Lack of Clear Error Messaging or Alert
Related Heuristic	Error Prevention
Proposed remedial (optional)	

Usability issue	Too Many Notifications
Related Heuristic	Minimalist Design
Proposed remedial (optional)	

STEP 5 – SUMMARIZE

- Highlight the **app's strengths and weaknesses** based on your analysis.

Strengths

-Grab has a simple and easy-to-use.

-Grab offers many different services to cater to various user needs. For example, Grab provides services like ride-hailing, food delivery,

Weaknesses

- Inconsistent Button Labels: action buttons (e.g., "Booking Tuk Tuk" for rides vs. "Place Order" for food delivery) can cause confusion.

- Suggest **specific improvements** to address identified issues.

Use a consistent label all confirmation buttons as "Confirm" or "Confirm Order" to avoid confusion.

APPENDIX – USABILITY HEURISTICS GUIDE (7)

HEURISTIC	**Visibility of System Status**
GOAL	The design should always keep users informed about what is going on
CHECK LIST	Are system status **updates** (e.g., success or error) provided **in real-time**? Is the user **informed of ongoing processes** (e.g., uploads, downloads, or syncing)? Do icons, animations, or visual cues **provide feedback** for user interactions (e.g., button presses)? Is there a clear **loading indicator** for actions like content loading?

HEURISTIC	**Match Between System and the Real World**
GOAL	The design should speak the users' language
CHECK LIST	Are labels, icons, and terminology **familiar** and aligned with the user's expectations? Is the user **familiar** with the terminology used in the design?

HEURISTIC	**User Control and Freedom**
GOAL	Users need a clearly marked way to leave unwanted action
CHECK LIST	Can users easily **undo actions**, such as deleting or modifying an entry? Is it easy to **exit or abandon** tasks without unintended consequences? Can users easily **interact with the inputs**, **buttons** using their **thumb** (mobile usage)?

HEURISTIC	**Consistency and Standards**
GOAL	Users should not have to wonder if different words, situations, actions mean the same thing.
CHECK LIST	Are design patterns (e.g., button styles, fonts, colors) **consistent across screens**? Are standard **platform conventions** followed (e.g., iOS vs. Android design guidelines)? Are icons**, gestures, and interactions** consistent throughout the app? Do **repeated actions behave the same** way every time (e.g., swiping left to delete)? Are similar features in the app named or **structured identically**?

HEURISTIC	**Error Prevention**
GOAL	Best designs carefully prevent problems from occurring in the first place.
CHECK LIST	Are potentially harmful actions (e.g., deleting an account) **confirmed** with a dialog or warning? Are **inactive or disabled elements visually** distinct to prevent misuse? Are actions **logically sequenced** to minimize mistakes (e.g., confirming details before proceeding)?

HEURISTIC	**Minimalist**
GOAL	Interfaces should not contain information that is **irrelevant** or rarely needed.
CHECK LIST	Are **primary actions prominently** displayed, with secondary options de-emphasized? Are **visual hierarchies clear**, guiding users toward their goals efficiently Are colors, fonts, and images used purposefully and consistently to **avoid distractions**? Is **negative space** effectively used to improve readability and focus?

HEURISTIC	**Help Users Recognize, Diagnose, and Recover from Errors**
GOAL	Error messages should **indicate the problem**, and suggest a **solution**
CHECK LIST	Are **error messages clear**, specific, and written in plain language? Do error messages suggest **actionable steps to** resolve the issue? Is the error's location **visually highlighted** for easy identification? Are users notified of errors immediately and not after they've progressed further? Can users easily **retry or correct** an action without starting over?