Five Major Tasks: Implementation

Here is a list of the 5 major tasks that were implemented in our Wildfire Billeting App.

A. **Searching for available billets (Robert):**
   This task is where we allow the user to search for a billet based on their desired specifications such as location, date and number of guests. A scrollable cardview result set has all of the available billeting options.

B. **Booking a billet (Oliver):**
   This task allows the user to book and pay for an available, selected billet for a specified time range and as such allows the user to view a summary of their requested billet and change any details before allowing them to pay for their booking.

C. **Hosting a Billet: (Tanner)**
   This task should allow the user to list their property/location in the app. Within this task, the user can create a listing and set parameters such as availability, description and price.

D. **Managing bookings: (Stefan)**
   Within this task, the user can view all of their bookings. From there, the user should be able to view, modify or cancel any existing bookings. From this task, viewing past bookings and leaving a review of the listing is also possible.

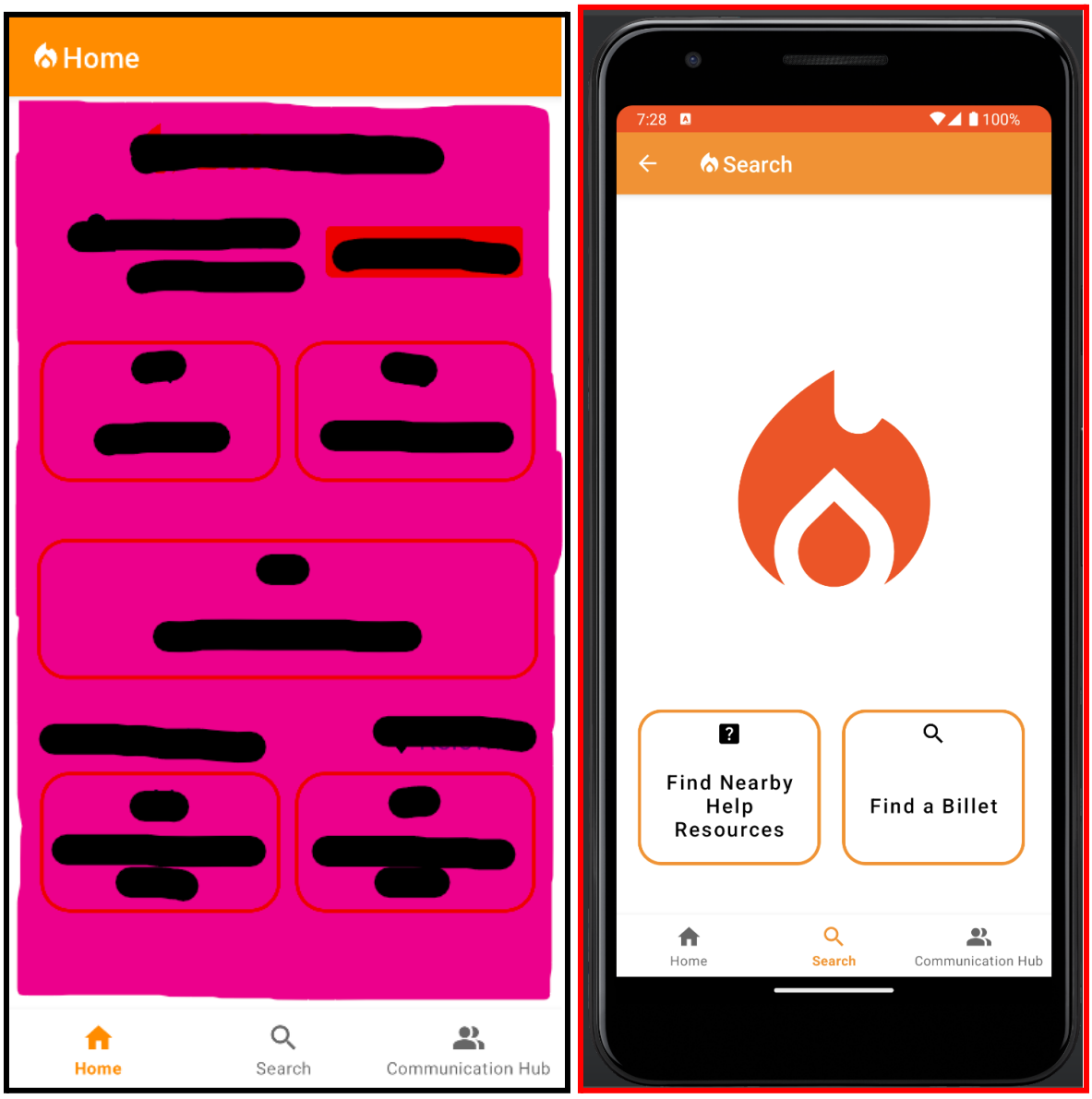
E. **Communication hub (Stef & Robert):**
   The communication hub task is where depending on the user, they can communicate with billet hosts/billet users of any upcoming or current bookings. Within the communication hub, there is a news section with important real-time updates from local official authorities with any pertinent nearby wildfire updates.

Design Principles:

Visibility (A):

A1. Tab bar style navigation menu improves system visibility as it is present throughout the entire user interface (UI). The current tab has its name displayed at the top, and other popular tabs are easily reachable from the bottom nav bar. (I have scribbled out everything except the navigation bars)

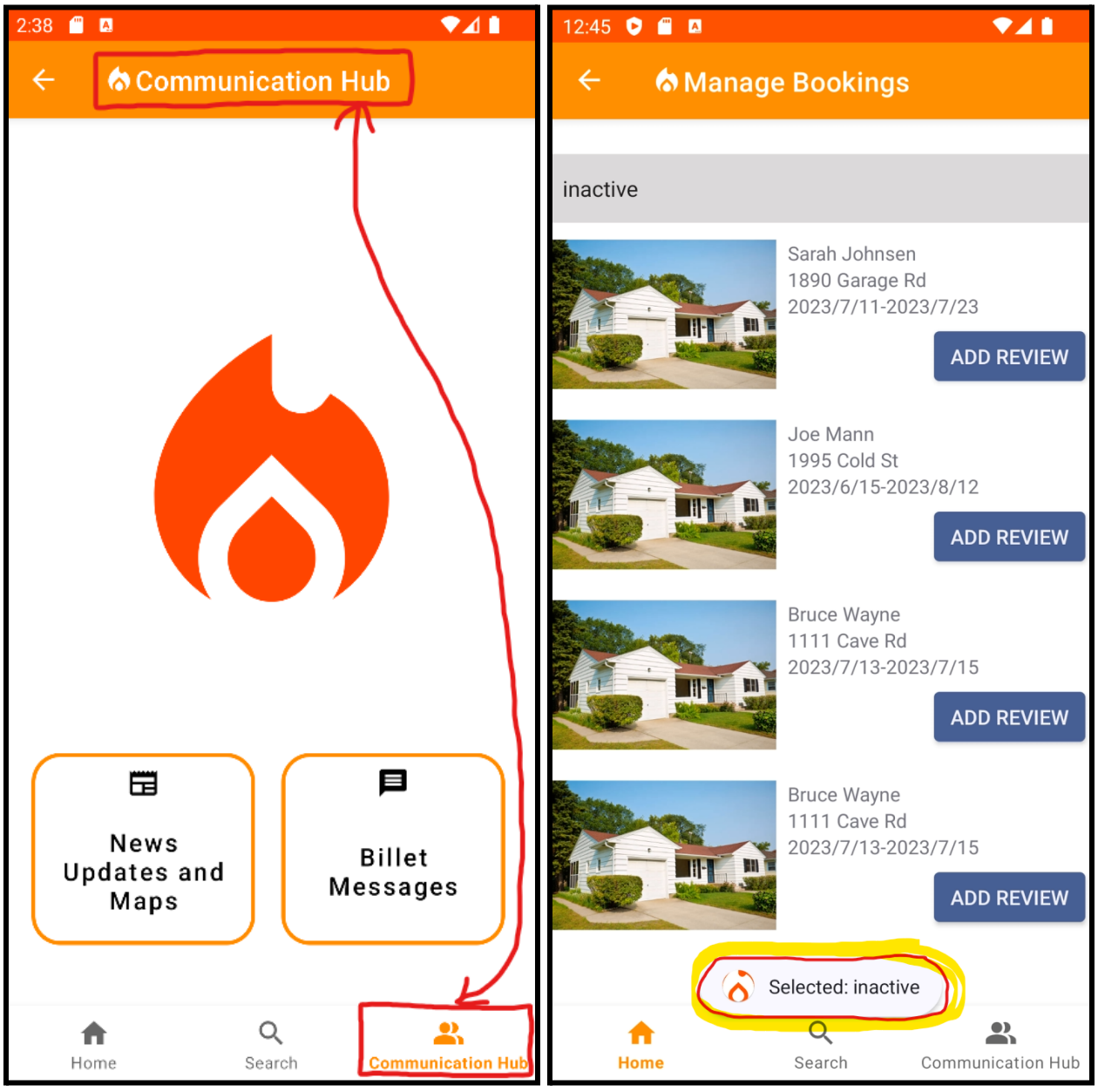
A2. Another example of visibility is seen throughout the UI with the back buttons in the upper left when in a fragment other than the home screen this button appears. It appears in the action bar which clearly indicates what it does and so the user knows solely by looking at it that this goes back to the previous screen and increases system visibility.



Feedback (B):

B1. The nav bar selection is highlighted depending on what tab the user is on. (For the following tabs: Home, Search, Communication Hub) *[POTENTIAL IMPROVEMENT by graying out all buttons if user is not on any of the three bottom (nav bar) tabs]*

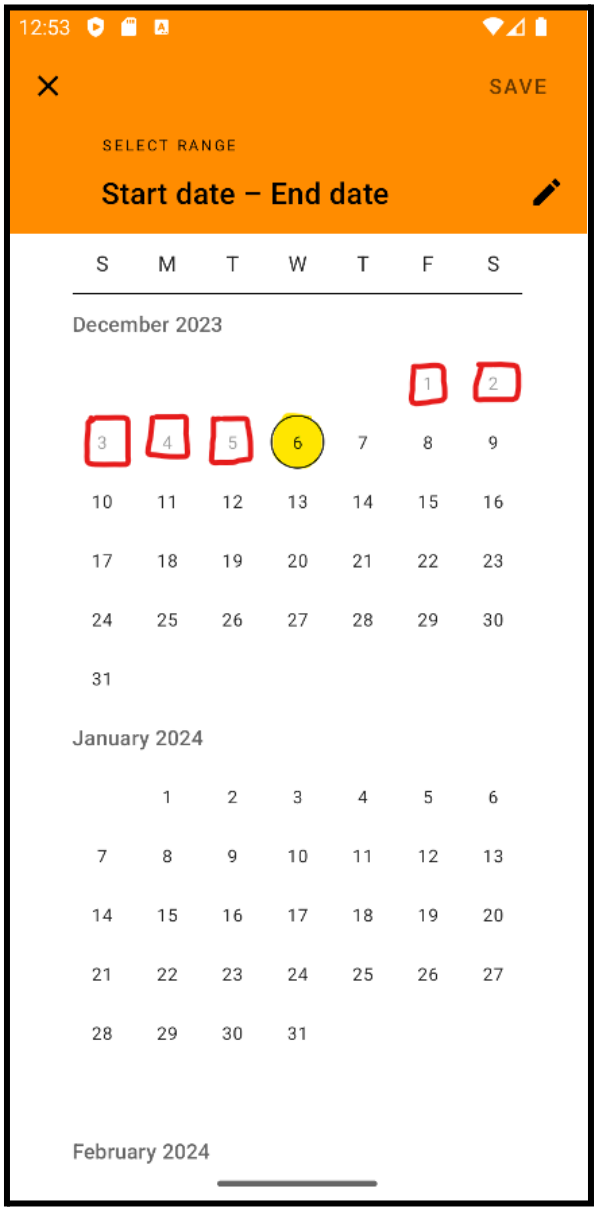
B2. In the Manage Bookings tab, a toast is displayed at the bottom of the screen when the user changes their filtering option. This toast informs the user that their filter choice has been updated, and states the newly selected setting.

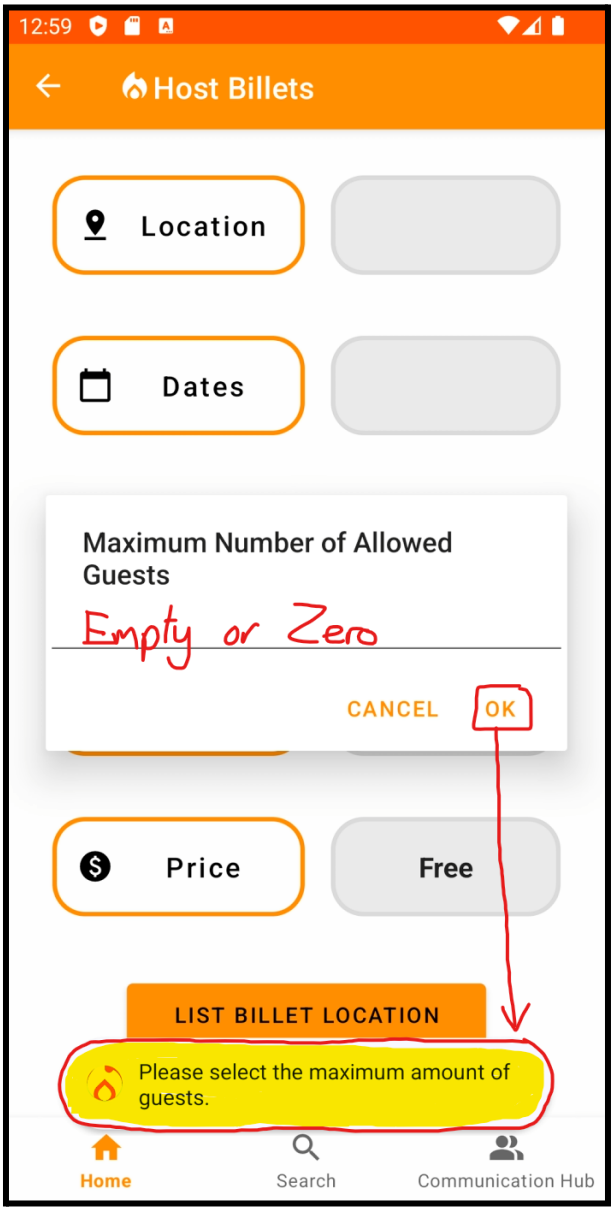


Constraints (C): *[Potential Improvement: graying out buttons until all data fields are filled]*

C1. When selecting a range of dates from the calendar interface, the user is prevented from selecting any dates in the past (boxed in red). The user is limited to selecting the current day (highlighted yellow) and/or a date in the future.

C2. When selecting the maximum amount of guests allowed at a billet location, if the user leaves it empty or enters a zero, an error toast will be displayed at the bottom of the tab. Having a maximum of zero guests would defeat the purpose of posting the listing.

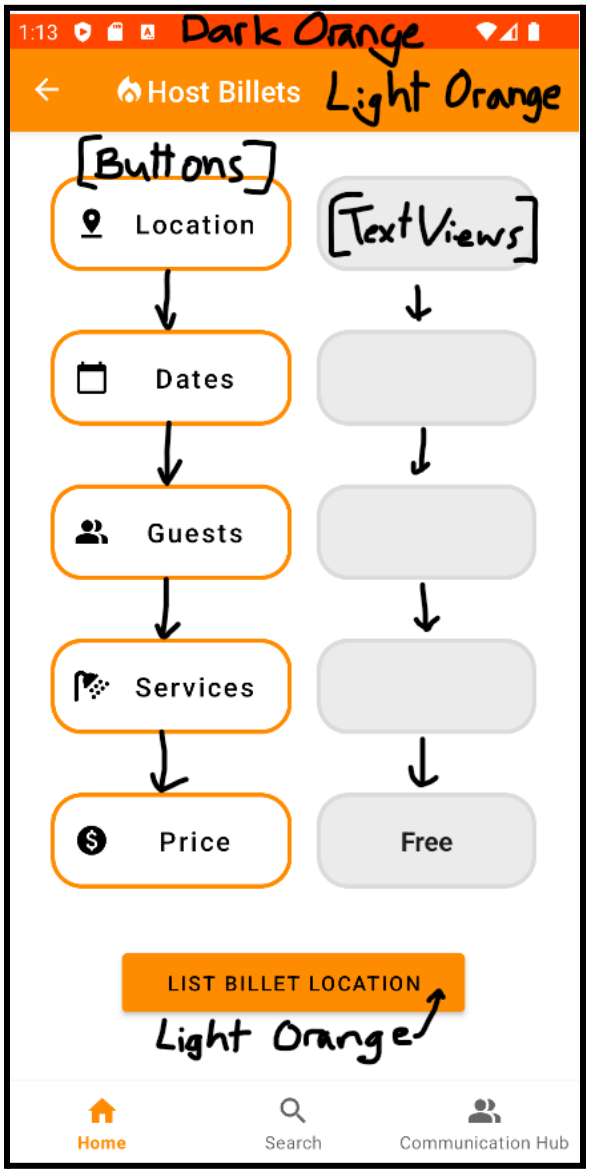




Consistency (D):

D1. Consistent theme across the entire user interface (UI), utilizing two shades of orange (light and dark) to colour the majority of our UI elements. Buttons and TextViews maintain consistency in their colours, form, and usage. Buttons are easily spotted by their bright orange border, and TextViews will often look almost identical to buttons but are grayed out with a less substantial border. This is an attempt to visually communicate which widget is interactable.

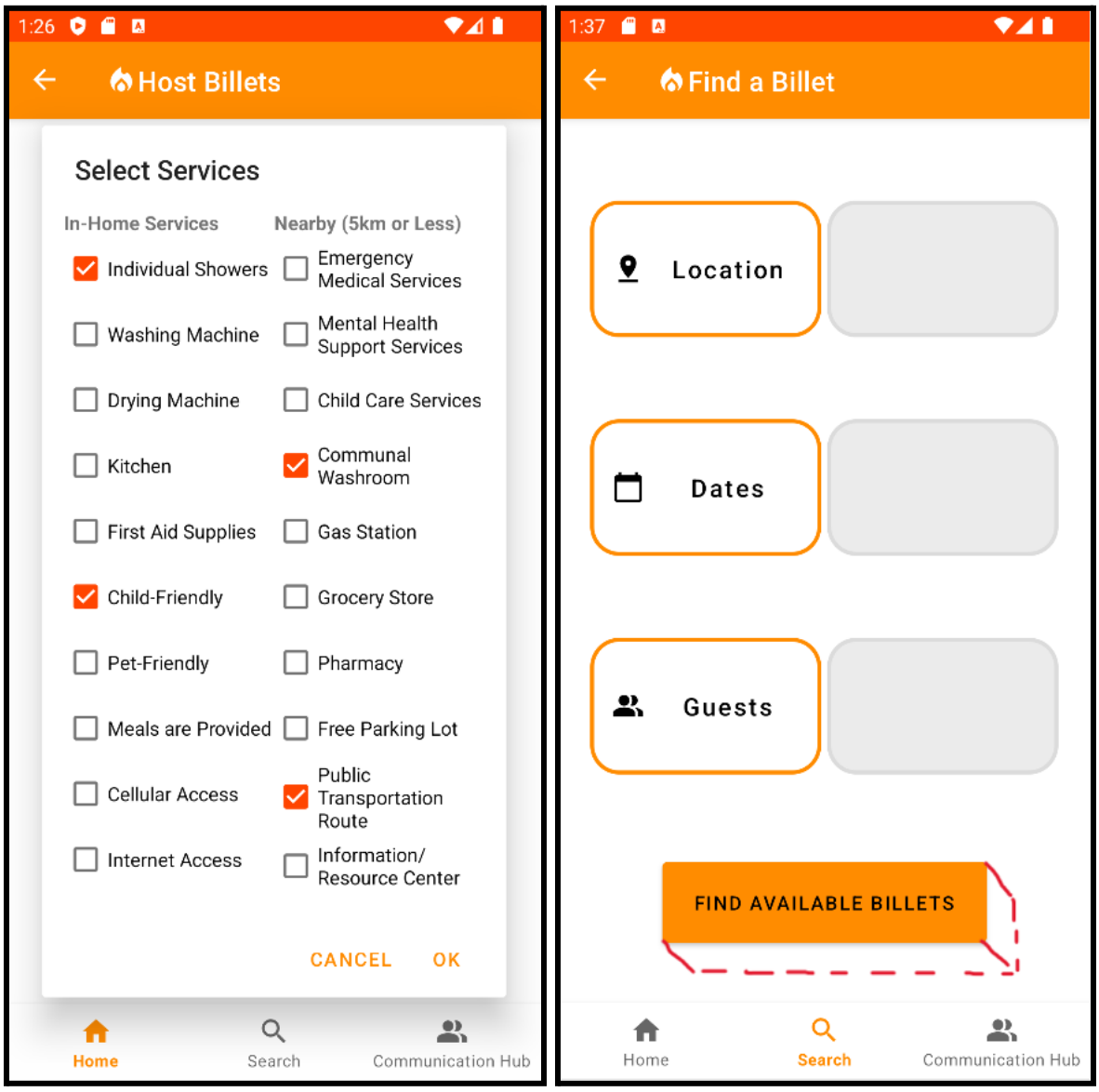
D2. Consistent logic throughout the app for selecting a location and picking a date range, both of those features were implemented using the same approach. The map and calendar interfaces are utilized in both the Search and Host Billet tabs. (To save space on screenshots, the calendar interface is shown in Figure C1 above)



Affordance (E):

E1. The Host a Billet tab utilizes a list of checkboxes for selecting services offered at the location. We use the square checkbox shape to communicate that multiple boxes can be selected at the same time. This is a common practice for a lot of software so it provides some external consistency as well.

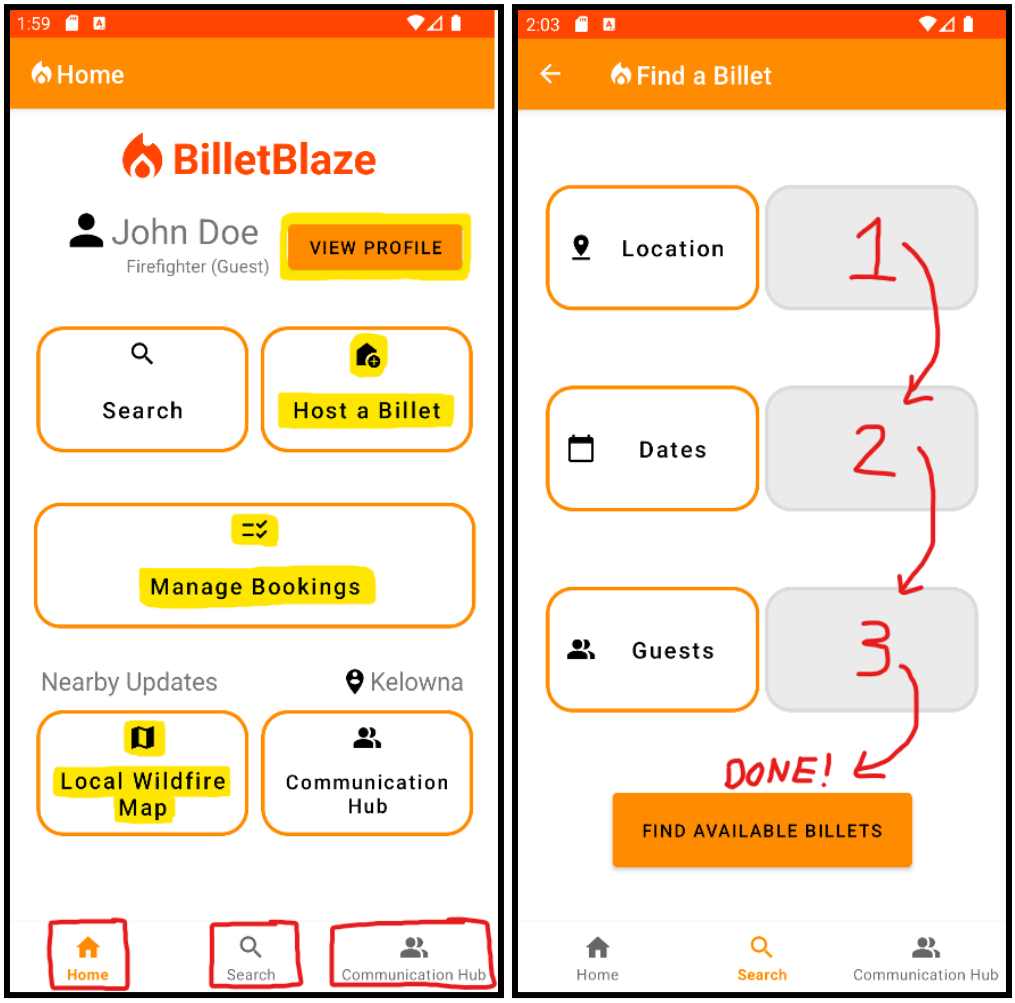
E2. Some of our submit buttons drop a very light shadow on the background behind them. (It may be difficult to notice in the small screenshot, the shadow is drastically extended in red for visibility purposes). This gives the buttons a three-dimensional look as if they are slightly raised off the screen. We combine this with a simple, straightforward caption to give the hint that it is a big button ready to be pressed into the screen. We also use vivid colour to manipulate the user's vision and draw attention to an important UI element.



Simplicity (F):

F1. The bottom navigation bar does a really good job of preserving simplicity. The nav bar only contains three tabs, Home, Search, and Communication Hub (boxed in red). No functionality is lost by not including other tabs because the Home and Communication Hub tabs host links to all other fragments not included on the nav (highlighted in yellow). They act as main hubs that can connect the user to wherever they need to go, hence why only those two tabs are ultimately required on the nav bar. Search is also handy to keep quickly accessible because most users will be interacting with the app primarily to browse and book listings.

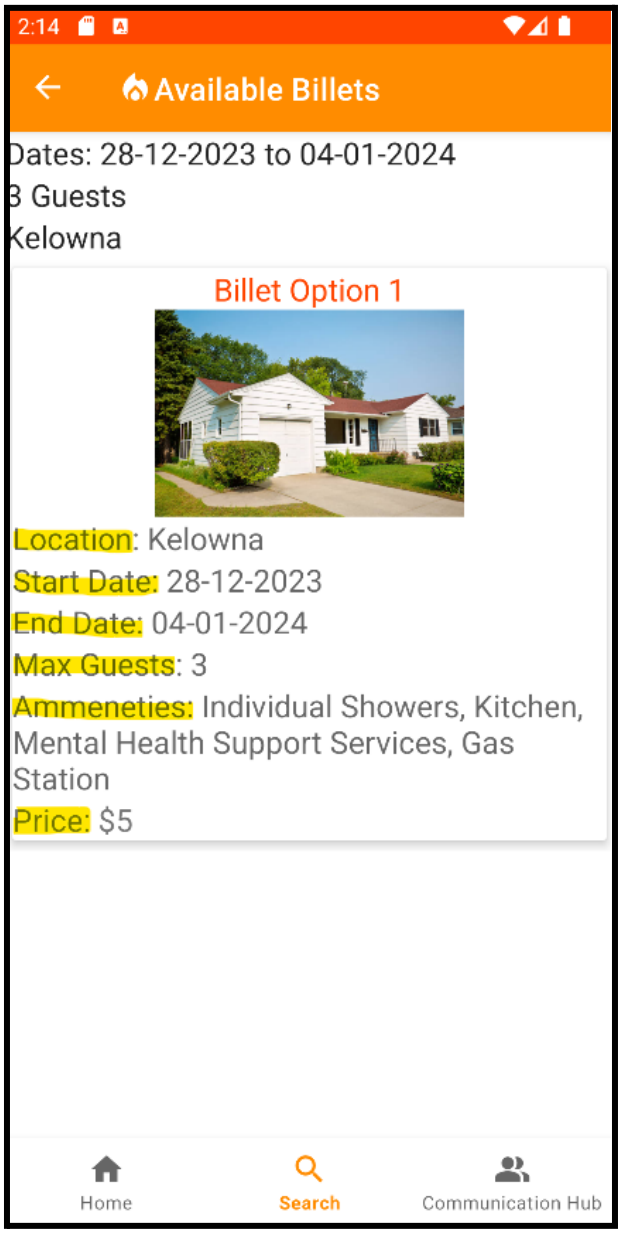
F2. The Search tab also helps to preserve simplicity. When hosting a billet, a user must enter a location, date range, maximum number of guests, available amenities, and price. The Search tab only asks for the location, date range, and number of guests. Making the search process quicker and simpler is beneficial to users who require immediate shelter. Further filters can be applied for those who have the luxury of time, but for those who do not, a broad array of listings will be returned as quickly as possible.

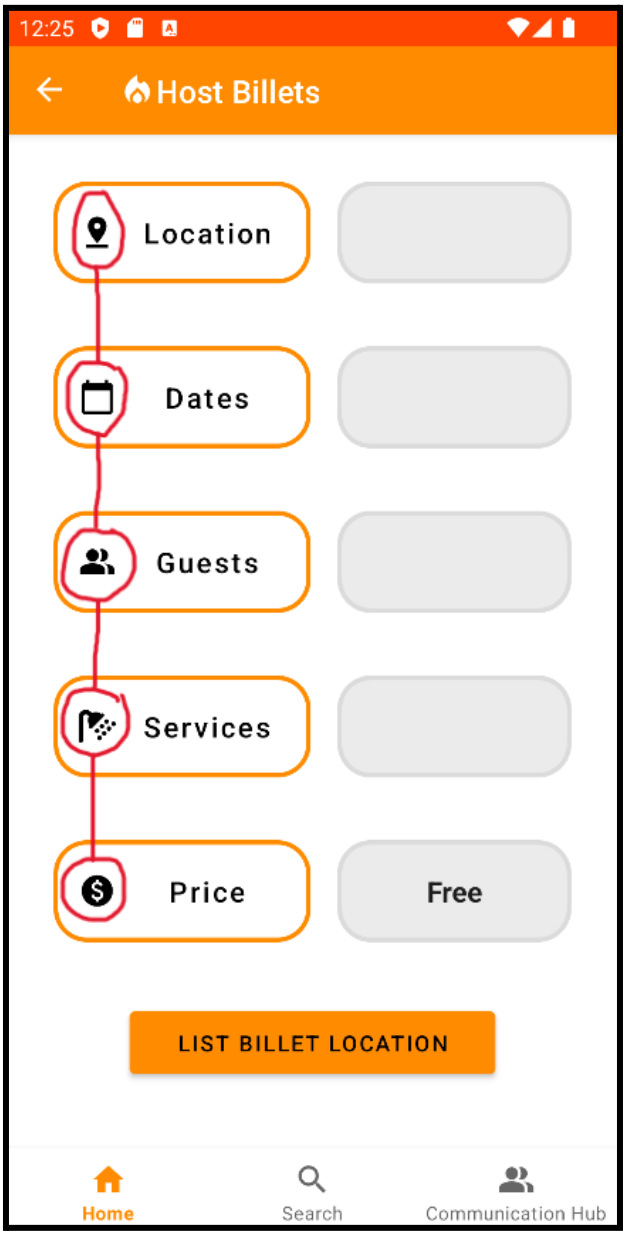


Matching (G):

G1. The Search functionality returns a list of available billet locations. These listings take data from an internal text file and display it in a plain and readable format. This process turns one line of comma-separated values into a visible advertisement for users to view. The system uses simple labels that match and follow a logical order rather than just displaying the internal representation.

G2. All button widgets utilize a small high-contrast icon next to their main text. These small icons help add context to their functions. For example, the “dates” has a calendar icon, the“offered services” button has a shower icon, and the “price” button has a dollar sign.





Help (H):

H1. When choosing a location in the Google Maps interface, (for Search or Host) a long toast is displayed that instructs the user how to choose a location. (Long click on the map to pick a city) This is required because the map interface has no other interactable elements.

H2. The payment screen assists users with how they should enter their payment info. For example, the CVV details on some payment cards can be difficult to find, and most users don't recognize CVV by the acronym. We provide a small hint on where to find the CVV details on most standardized payment cards.

A summary of problems identified by each user:

You can sort these in terms of severity, functional/conceptual area (i.e., in a way that makes sense concerning your system), or each heuristic.

Violation: Visibility

Severity: 1

Problem: The contacts and messages were displayed so that they seemed like they belonged to the same object. Text views were too close together and the user could not tell which item was which.

Recommendation: Re-evaluate XML design

Violation: Consistency and Standards/Aesthetics

Severity:1

Problem: When on the search screen after the user selects a location, date range and amount of guests the info displayed was only displayed as a plain text view without any styling.

Recommendation: In the host a billet section the textviews are displayed as boxes next to the buttons. To be more consistent and increase the visual appeal of the app do something similar to that.

Violation: Error prevention/Standards

Severity: 2

Problem: The map for search does not have a “locate me” button and when opening the map it is set to somewhere in Africa and not in Kelowna where the app is primarily used so there are misclicks when wanting to select a location which results in a “slip” and can be prevented with good error prevention and matching to how the user would expect to use a map app.

Recommendation: Add a locate button.

Violation: Consistency and standards

Severity: 2

Problem: The keyboard overlapped the text input so the user could not see what they were typing in the messages to the billet host.

Recommendation: Format the XML layout so that the text input goes up with the keyboard as well as the message recycler view.

Violation: Error prevention
Severity: 3
Problem: The user was unable to enter their credit card information as they could not figure out how to properly format the credit card. Originally the user entered “4444444444444444” which is a 16-digit long character, but the app wanted the user to enter “4444 4444 4444 4444”.
Recommendation: Auto spaces or toast and hints.

Violation: Consistency and Standards
Severity:3
Problem: Originally, the app only had one billet that was filtered by any results that the user entered. To fix this we allowed the user to create billets based on certain specifications which they could then search for based on certain filters. Now, only billets that met the filtered specifications would then show in the results.
Recommendation: Add a scrollable list of all billets that were posted via host a billet.

Violation: User control and freedom
Severity:3
Problem: After the user uploads a billet via host there is no way to go back home.
Recommendation: Add a “bring me back home button” on the dialogue not only a dismiss.

4. Modifications

Report should discuss design fixes made based on the heuristic evaluation in another iteration.

- We created a hint with the correct formatting for the credit card number entry and a toast message that tells the user what the proper format is.
- We created functionality for the user to add listings and specify their search.
- Displayed the text in the search more aesthetically. To be more consistent with the design, in the host, a billet section the views are displayed as boxes next to the buttons.
- We added a dialogue after the user posts a billet to go back home or dismiss the popup dialogue in the host section.

- In the messages, we made the text view go up on the screen so the user's text is visible while the user is typing a message to the host or vice versa.
- In messages, implemented a card view into a grid layout and created bubbles for each object so the user could recognize which was which (messages/contacts) and that they were clickable objects.
- We added a "locate me button" for the maps and we set the location to Kelowna when the user opens the map to select a location.
- We put the contact and message lists into a card view grid layout and created bubbles for each object so the user could recognize which was which and that they were clickable objects.

5. Updated prototype:

Updated prototype based on the heuristic evaluation

-------[Revised changes mentioned above were all implemented in the Billetblaze app]----------

7. Video link showcasing all five tasks:

A video showing how you expect your system to be used for the five tasks

https://www.youtube.com/watch?v=2pZWxd4pR0A&feature=youtu.be&ab_channel=OliverMedgyesi