GitHub Username: jainakshansh

Kwik

Description

Interfacing an ad-hoc social event planner that provides individual assistance for attendees of a group social event such as night out, a movie, or a dinner.
Kwik focuses on solving the overall event problem, focusing on easy event creation and providing users with utility functionality.

Intended User

The application's primary users can be classified in two -

a. User's opening the app to plan events with ideas such as - "What can I do today?" or "How can I spend my free time tomorrow?" - to figure out a way to get rid of their boredom.
b. User who opens the app in order to create and plan an event - that the user have already thought of, thus using social interaction features to carry out the event idea.

Features

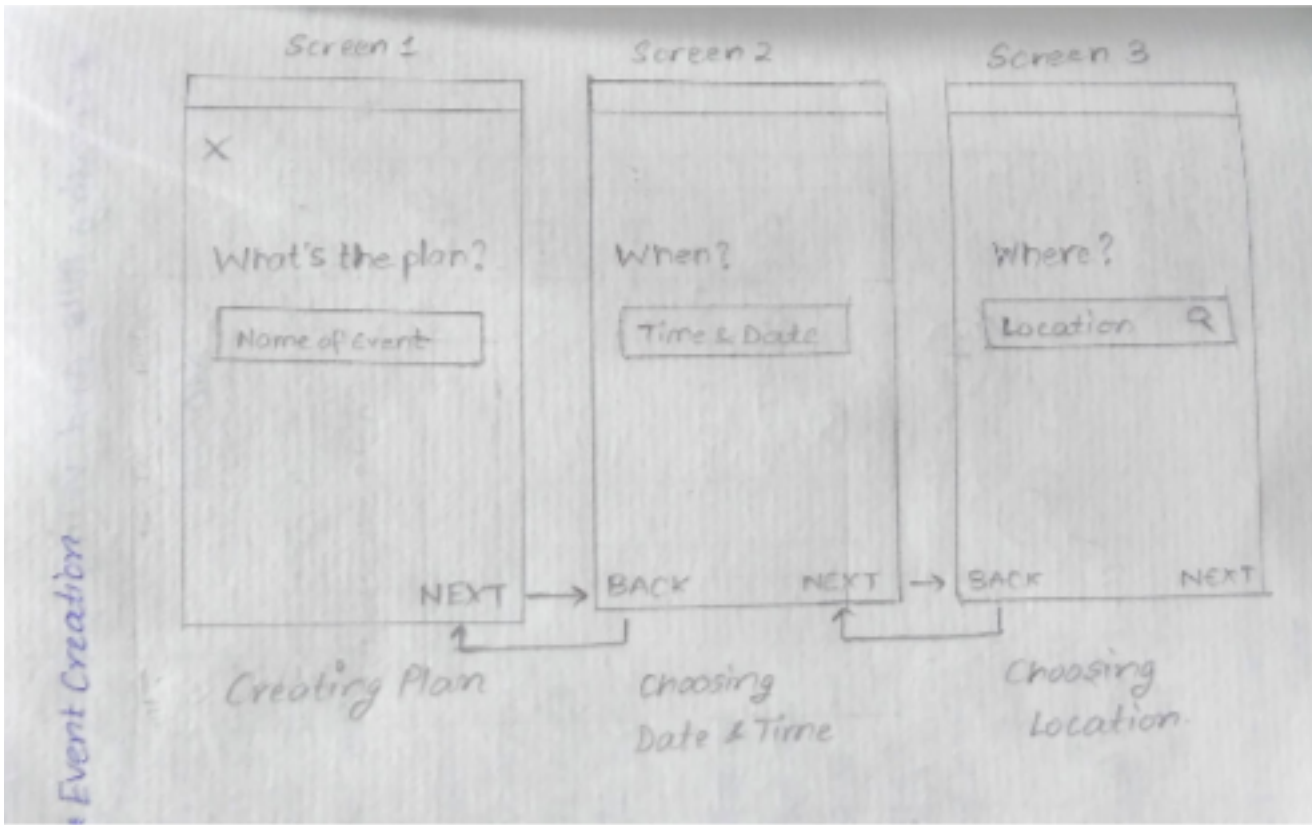
- Easy and intuitive event creation.
- Event Details page - with general details, participants, etc.
- Easy polling feature to reduce decision fatigue.
- Utility features like - reminders through calendars, navigation through maps.
- Sign In feature - using Gmail ID with Firebase as a backend for data persistence.

User Interface Mocks

Screen 1 - Landing Screen

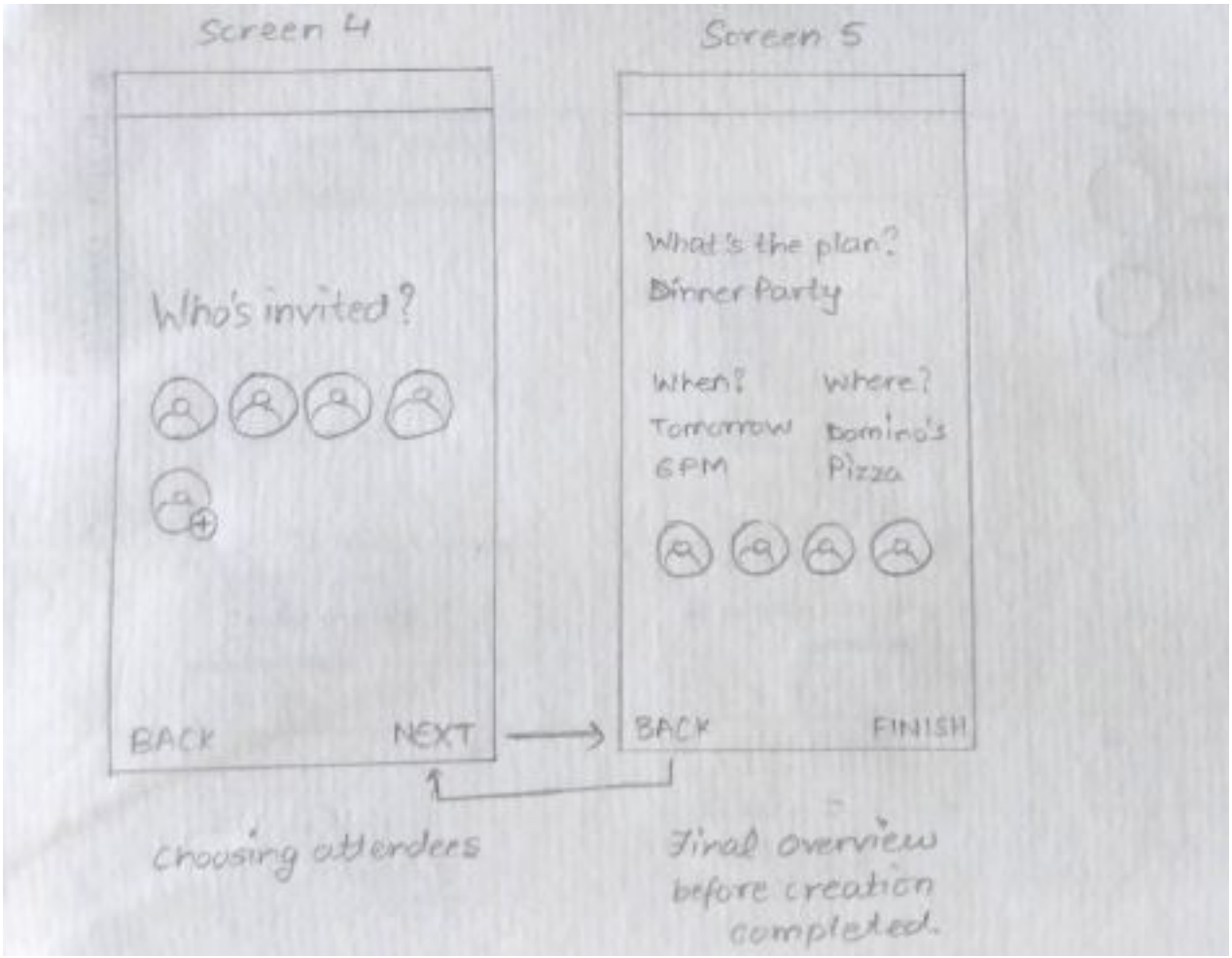
This is the landing screen. I'm aiming of using a Bottom App Bar with a Floating Action Button which acts as a Call-To-Action (CTA) to create events. The app bar contains a Navigation Drawer and another button for visiting User profile. The screen will contains cards, each depicting one upcoming event to the user.

Screen 2 - Event Creation

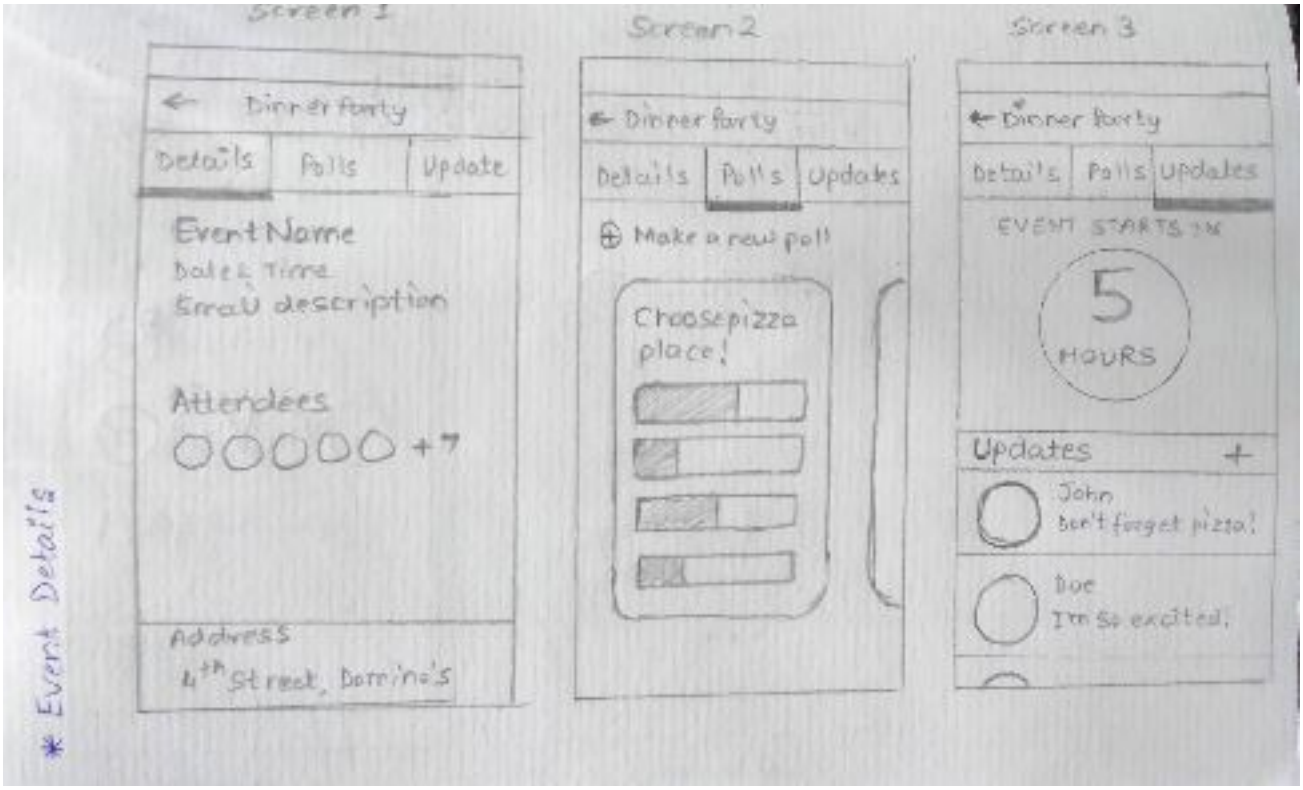


These screens focus on the event creation flow -

a. First one asks user for the Name of the Event or Plan.
b. Second one is focused on getting the date and time of the event.
c. Third one is to select the location of the event.
d. Fourth one asks the user about the attendees of the event.
e. Last one gives an overall summary of the information entered before user confirms the event.


Screen 4
Screen 5
Who's invited?
BACK
NEXT
What's the plan?
Dinner Party
When?
Tomorrow
6PM
Where?
Domino's
Pizza
BACK
FINISH
Choosing attendees
Final overview
before creation
completed.

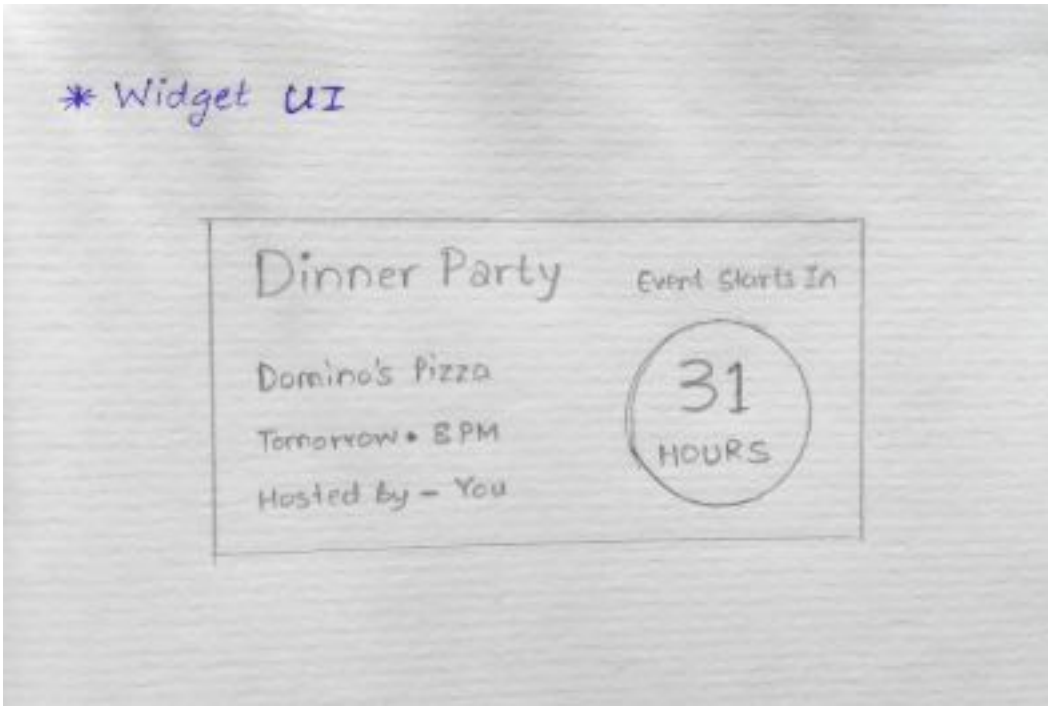
Screen 3 - Event Details



These screens depict the actions that can be performed after an Event is created.

a. The first tab “Details” reveals all the detailed information to the user about the event.
b. The second tab “Polls” allows attendees to create polls. This can be helpful in choosing things like “location of the event” or “deciding on the time”, etc.
c. The third tab “Updates” allows users to send update notifications to other attendees so as to reduce the need for a chat interface to discuss things.

Screen 4 - Widget Mock UI



The widget will show the latest upcoming event/plan the user has created or is interested in. This also includes an overview of the Location, Date & Time, Host Name, and ETA to the event.

Key Considerations

How will your app handle data persistence?

I plan to use the Firebase Realtime Database - which can store and sync data with all the users who will have logged in through their Gmail ID.
The sync functionality will include -

a. Updating the events they are going to or interested in.
b. Updating the users from their contact list.
c. Updating the event details related to the user.

For offline functionality, I am aiming to use Room Database to store a part of the event along with the detailed information.

Describe any edge or corner cases in the UX.

- Since the app doesn't incorporate or uses Maps to determine location, phrases like "My Place" can be a problem for attendees who are acquaintances, for example.
- Since the invites are through a Gmail ID, users who don't have a Gmail ID but are connected through phone numbers, will have to create an ID to use the application.
- In case of unstable or missed network connection, the application must not crash.
- The application should be able to handle Device Orientation Change correctly.
- The application must not perform any demanding operation on the main thread which might cause the app to feel buggy or freeze.

Describe any libraries you'll be using and share your reasoning for including them.

- Picasso or Glide - to handle image loading.
- Firebase libraries - including Authentication and Realtime Database.
- Material Intro View - showcase library to highlight important UI elements during first use.

Describe how you will implement Google Play Services or other external services.

Google Play Services -

- Firebase Sign In
- Firebase Realtime Database

Required Tasks

Task 1: Project Setup

Creating a setting up a new project from scratch. This will include -

- Creating a new project in Android Studio.
- Configuring the required libraries.
- Initializing the Github project repository.

Task 2: Linking the app to Firebase

For this task, I will -

- Creating a new Firebase project.
- Using SHA keys and google.json files to sync the project with Firebase.

Task 3: Creating the UI

This will include -

- Starting development w.r.t. The UI mocks.
- Focusing on responsive design.

Task 4: Building the app logic

For this task, I will -

- Setting up Firebase Authentication within the app.
- Setting up the Firebase Realtime Database within the app.
- Test run the services above to make sure they are working fine.
- Binding the view containers with the incoming data.

Task 5: Implementing Data Persistence

In this stage, I will explore different ways to make sure that data to some extent is available for offline use too.

Task 6: Notifications

Implementing Notifications which will notify users when an event they are attending or interested in is near.

Task 7: Developing the app widget

Creating the application widget, which will show the latest upcoming event the user is interested in or is attending. The user might be able to add a specific event to the widget too.

Task 7: Making the app production ready

This task will include the following steps -

- Providing a finish to the app design.
- Providing accessibility support.
- Creating and using app icons.
- Removing debug logs and messages.
- Generating the required keys and files (JKS).