GitHub Username
apelsoczi

Vendship

Description

Vendship is a Vendor Management app, that acts as a mechanism for users to source services and simplifies managing multiple vendors.

Intended User

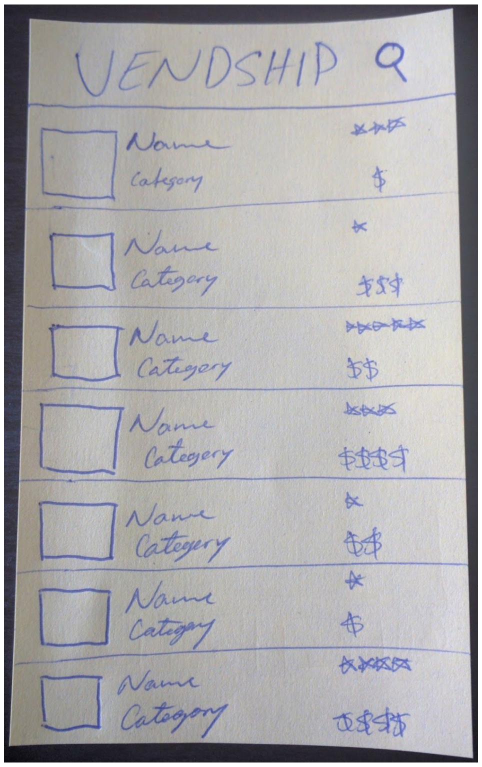
The intended user would be a party with a requirement to coordinate multiple vendors, such as for planning an event requiring a location, entertainment, services, suppliers, rentals and various labourers.

Features

The primary features of the app will be:

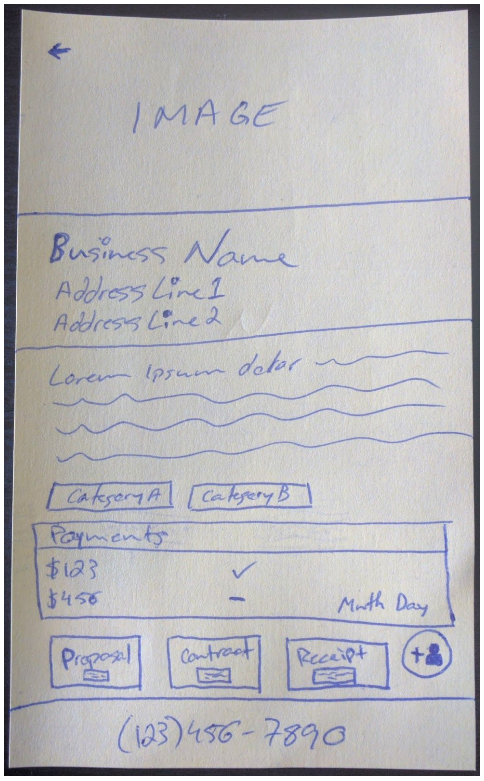
- Procurement
- Contacts
- Reminders

User Interface Mocks



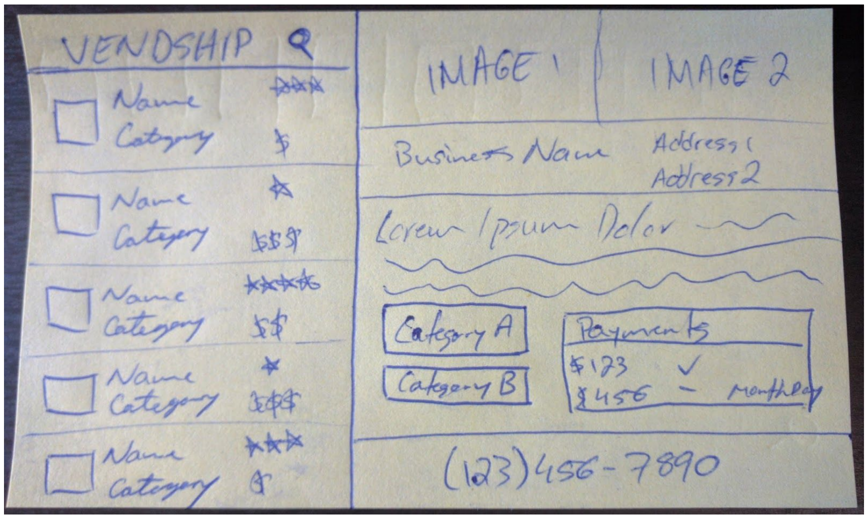
Vendor Screen

The vendor screen will be the main page for the user, users will be able to search the Yelp API for vendors based on the city of preference. This screen will be a fragment showing search results and favorited vendors.



Vendor Details

This detailed screen allows a user to view all of the information returned by the Yelp API for individual business'. User's will be able to share share information such as email, address and phone with other applications installed on their phone.



Tablet Layout

This will be a standard master detail flow composed of the Vendor Screen and Detail Screen.

Key Considerations

How the app handles data persistence.

App will implement a ContentProvider and also use features of the Yelp API to cache and request data in compliance of the Yelp Developer Terms of Usage. Data will be updated on a per user request basis using an IntentService. Other

UX corner cases.

This is a simple list / detail flow for user interaction. There are no real corner cases to be expected on phones. Activities and Fragments will be expected to be aware of tablet layouts and will be crafted to function on either phone or a master detail flow.

Libraries to be used and their advantages.

Picasso Library will be used to display images if returned from the Yelp API, eliminating the amount of boilerplate code required to create functionality. The Yelp API library also uses Retrofit to minimize redundant code.

Google Play Services to be Implemented.

- Location will be used, allowing the user to either search Yelp for business' by their current location or by one defined by user preference.
- Analytics will be used to understand user engagement allowing future releases to turn insights into action.

Required Tasks

This is a decomposition of the tangible tasks to complete incrementally to create a finished app from inception to execution.

Task 1: Project Setup

Architect Project base

- Create empty Android Studio project
- Create Application class as a singleton, themed to present a splash screen drawable while the application is loading.
- Setup Application module build.gradle with libraries required, sdk versions (min, target, compile) and import Google services.
- Extend AppCompat as the base theme to use Material Design components.
- Manage dependencies with Gradle.
    - Support Library, Google Services, additional Libraries.

Task 2: Implement UI for Each Activity and Fragment

Build the UI for multiple Activities and Fragments

- MainActivity Activity
    - Vendors Fragment, API search results and previously favorited Vendors.
    - VendorsDetail Fragment, response metadata for each Vendor from API.
- Settings Activity
    - Preferences Fragment, preferences the user interacts with customizing the functionality of the UX.
- Use standard and simple transitions between fragments and activities.
- Enable RTL Layout switching on all layouts.

Task 3: Implement Yelp API Library

Fetch Data from Yelp API

- Create a Yelp Developer Account and sign up for an API Key, include app module build.gradle defaultConfig.
- Create Parcelable POJO classes to model extracted data from returned data.

Task 4: User Experience

Add functionality for platform UX standards

- Allow users to share data with other applications
  - Vendor Email, Vendor Phone, Vendor Address, Vendor Website.
- Favorite Vendors
  - Implement a content provider to save vendors in a local collection

Task 5: Productionize from a functional state

- Validate data from servers.
- Handle error cases in user input.
- Include support for accessibility:
  - Content descriptions, D-Pad navigation.
- Allow for localization and distribution in global geographies.
- Add a Widget to proactively display upcoming reminders.

Task 6: Building with Gradle

- Build and deploy using installRelease Gradle task.
- Equip app with signing configuration.
  - Keystore and Passwords are included in Git.
  - Keystore is referenced by a relative path.
- Build app from a clean repository checkout without additional configuration.