**هات و خد**

**غير فكرك**

**Problem**

People with skills or education or hopies want to work as freelancer or practise this skills found it hard to take a job without professional experiences because the freelancer site takes advantage of the freelancers with money so they can wait for lang time before they reach a high level of exprenies

**Idea**

The idea is to introduce an environment where people can use their talents by giving services to others and take another service in return or get points which they will use to get services from others when they want. This will help them improve their skills as they will practice their talents and will help them benefit from many hidden talents and things which can help others even if this services are too small and easy but perhaps someone need them. The services can be: \*Helping someone make his timetable. \*Make logo for someone. \*Help someone improving his english. \*Help someone deciding what to wear. Or any service any one can imagine.

**Advantages:**

* Easy online marketplace for freelancers and hobbyists to cashlessly exchange services.
* Save time and efforts when you want to use other’s service.
* Overcoming the problem of high prices
* Support the Symbiotic Economy
* supporting small-scale entrepreneurs to do the flexible work they love.
* ideal for medium to large companies where employees don’t all know each other.
* Get more experience in the field

**Challenge**:

* Lack of trust in people

solation: provide them with two options. You can say: *“I’m interested in getting a massage from you, I’d be willing to exchange [your service].*

*or I can compensate you with that application.*”

Providing options makes more likely to find a match. And if you do get a “no” — then ask someone else!

* difference of cultures

**Technologies used**

1. Server Side
   1. MySql DBMS System
   2. Hibernate O.R.M System for its many advantages
   3. Spring Web Service
   4. Spring Security
   5. Spring Data
   6. Spring MVC for WebView
   7. Jsp Web Pages
   8. Google Cloud Messaging
   9. AOP programming
   10. Maven project management
   11. Github VCS system
2. Mobile Side

### ButterKnife UI object injection

### Picasso Image Loader and cacher

### GSON json converter from json to object

### Retrofit http request handler and chacher

### Event Bus handle the data transfer from Activity or fragment

### ActiveAndroid ORM System for sqlite

### LeakCanary help detect memory leaks for your app

* 1. RecyclerView prevent memory consumption when viewing list
  2. CardView help create good UI view
  3. Google Cloud Messaging for Server client Push notification
  4. Constraint layOut the latest Layout with reactive design
  5. Gradle project management
  6. Github VCS system

**Some limitation of these technologies are:**

* Developing native mobile apps:
  + Cost of development. Native applications are more expensive in development than cross-platform and web applications.
  + Intercompatibility with other mobile OS. Native apps for iOS are developed using Swift or Objective-C and can be installed on iPhones or iPads only. Likewise, Android apps aren’t compatible with iOS because they’re built with Java or Kotlin. Another aspect that is worth mentioning is version compatibility. For example, because of a variety of Android versions, providing full support to all the app users becomes a real challenge.
* Using java as our developing language in both front and back ends:
  + Slow performance: The first and foremost drawback everyone accepts is Java's slow speed of execution.   
    No support for low-level programming: With Java, low-level programming cannot be done as it is intended to be portable across platforms.  
    No control over garbage collection: Garbage collection is one of the built-in features of Java and is entirely managed by JVM.
  + No pointers: low control over memory (RAM)
  + High cost of servers and licences over server side.

**Main Functions for users**

1. User can register with social media like facebook or gmail ,etc.
2. Anonymous user can view and search services and requests but can't apply for them.
3. Filtering the search data with names ,location , category,etc.
4. Logged in user can apply for request and post service.
5. When posting service, he can assign availability for it and price in points and type of payment.
6. Payment type can be with points or anther services.
7. User can chat with another at any time
8. User have a wallet of points that he use to get services
9. User can post request with the price he want
10. User can view all the requests and services he working on
11. User has history that represent all his services and his request
12. User can delete services or request only if he doesn’t have commitment
13. Only user how used services can submit reviews on the service
14. User can rate the skill of the one how gave him services
15. User can follow other users and can connect with the other friends on different social media

**Main functions for admin**

1. Admins can have different types and clearance
2. Admin can view user date and history
3. Admin can block,set penalty and more on users
4. Admin can create challenges and set the rules for it
5. Admin can give users more feature
6. Admin can see the user data with details like time and the relationship of.the users
7. Statics of the system is viewed
8. Admin can chat with the users if needed
9. Admin actions is.monitored as needed
10. Admins is review by other Admins and user

**Keep in mind**

**Service** is the user how can do something like making android app

**Request** is the job that some user want someone help with it

Anonymous user is someone who isn’t logged in

User refers to logged in user

**MainViews**

1. **Home Screen**

1) List Of Available Services Or Request

2)Search By Name,Category Location ,Username,Availability (From D,Y,T To D,Y,T)"Imdb"

3)List Of Categories (Availability Or Request)

4)List Of Top Users.

5)List Of Challenges And Availability Gifts.

6) Header \*Notification \*User Name \*Massages \*Add Service \*Add Request \*Sign Out \*Setting

1. **Profile**

1)Full Name ,Address Phones ,E.Mail And Other(Edit & Delete

2)Skills (Add, Edit ,Delete)

3) Edicatin (Add ,Edit ,Delete)

4)Social E.Mail Like Facebook & Gmail /Professional

Like Github ,Behind, Linkedin

5) Certification Verification (Further)

6)Availability

7)History Of Request And Services

8)Massages

9)Services (Edit Delete)

10) Privacy Customization (Further)

Add ,Edit Delete ,For User Others Can View Data

1. **\*Request Detail**

.Detail Information About The Service

.Mainfield Version Of Profile

1. **\*Service Detail**

.Detail Information The Service.

.Review & Comments

.Minified Version Of Profile

1. **\*Service Review**

.Skill Used Rating It

.Review Job Quality

.Reate Service .

Submit

1. **\*Search Result**

.List Of Generic Object

.Request (Apply),Service (Request) Or Users