Home Maintenance App

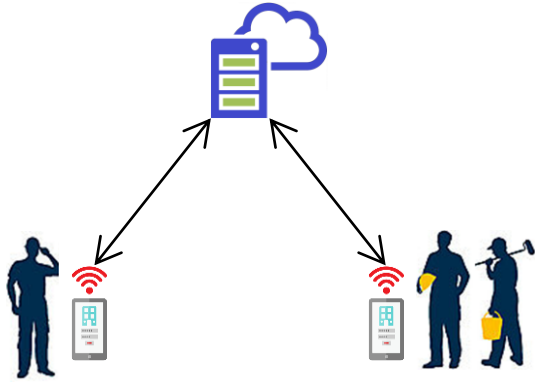
Prashant Patil

Problem Statement

- Develop an iOS/Android hybrid mobile application that will help people/consumers to solve day-to-day home based needs like calling an electrician, plumber etc.
- Use JS/HTML5 to build the app and also utilize any popular JS framework.

Functional Requirements (Processing Sequence)

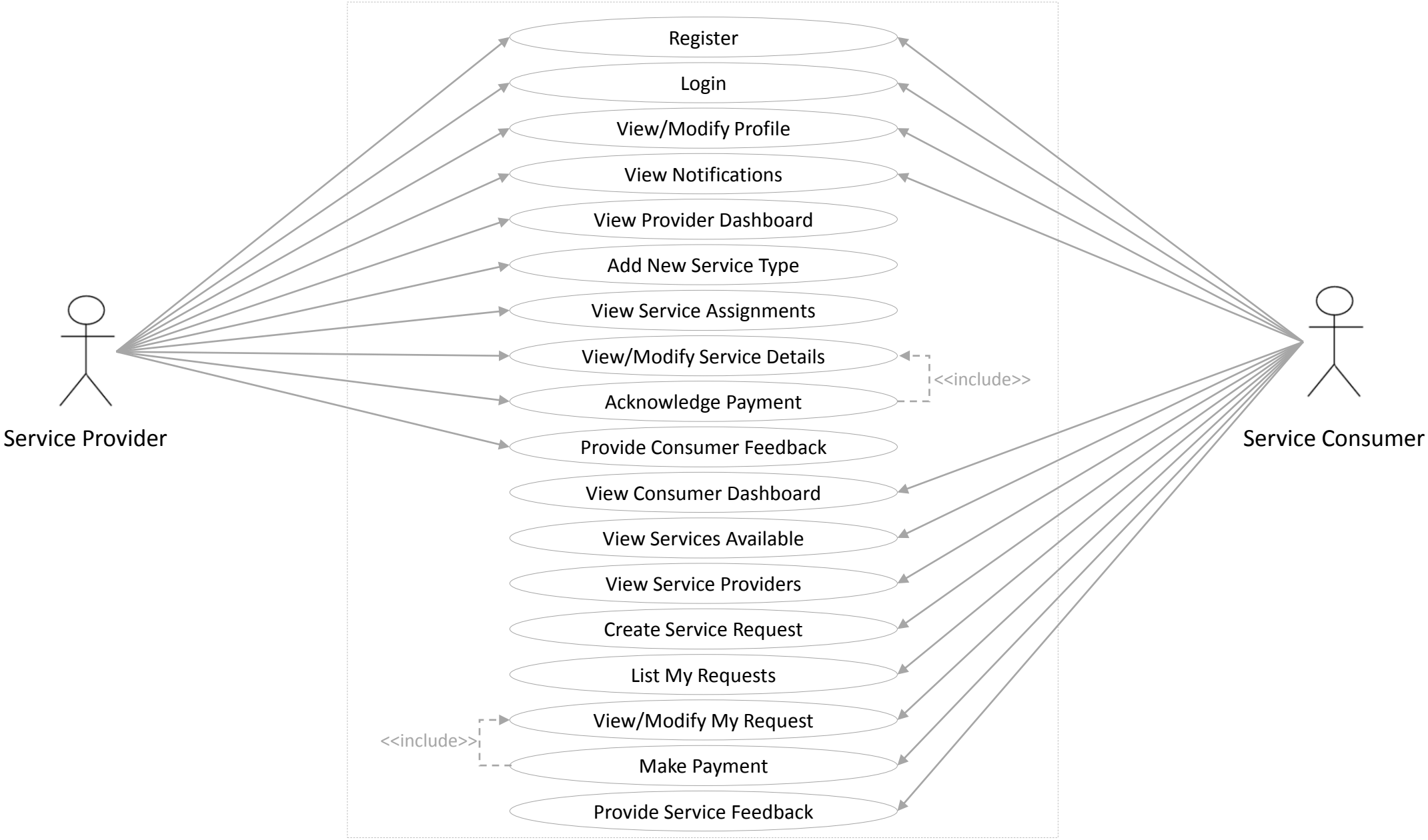
1. Provider registers, and publishes types of services being offered.
2. Consumer registers, and searches for services, based on location / type of service.
3. Consumer selects a service provider, and provides details of service request.
4. Provider receives notification, views the service request, and accepts/declines it.
5. Provider can locate / call the consumer, to reach the place.
6. Provider services the request, or marks it for follow up, or declines it.
7. Consumer accepts the resolution, makes a cash payment, and provides feedback.
8. Provider acknowledges the payment, and provides feedback for consumer.



Assumptions

- The application will be implemented in English language only
- The application will be available on the following platforms:
  - Android: version 4.0 and above
  - iOS: version 6 and above
- The application will work in portrait mode only
- The applications will be deployed in Apple and Google app stores

Use Case Diagram



User Interfaces

Common UI

- User Registration
- User Login
- User Profile
- Notifications

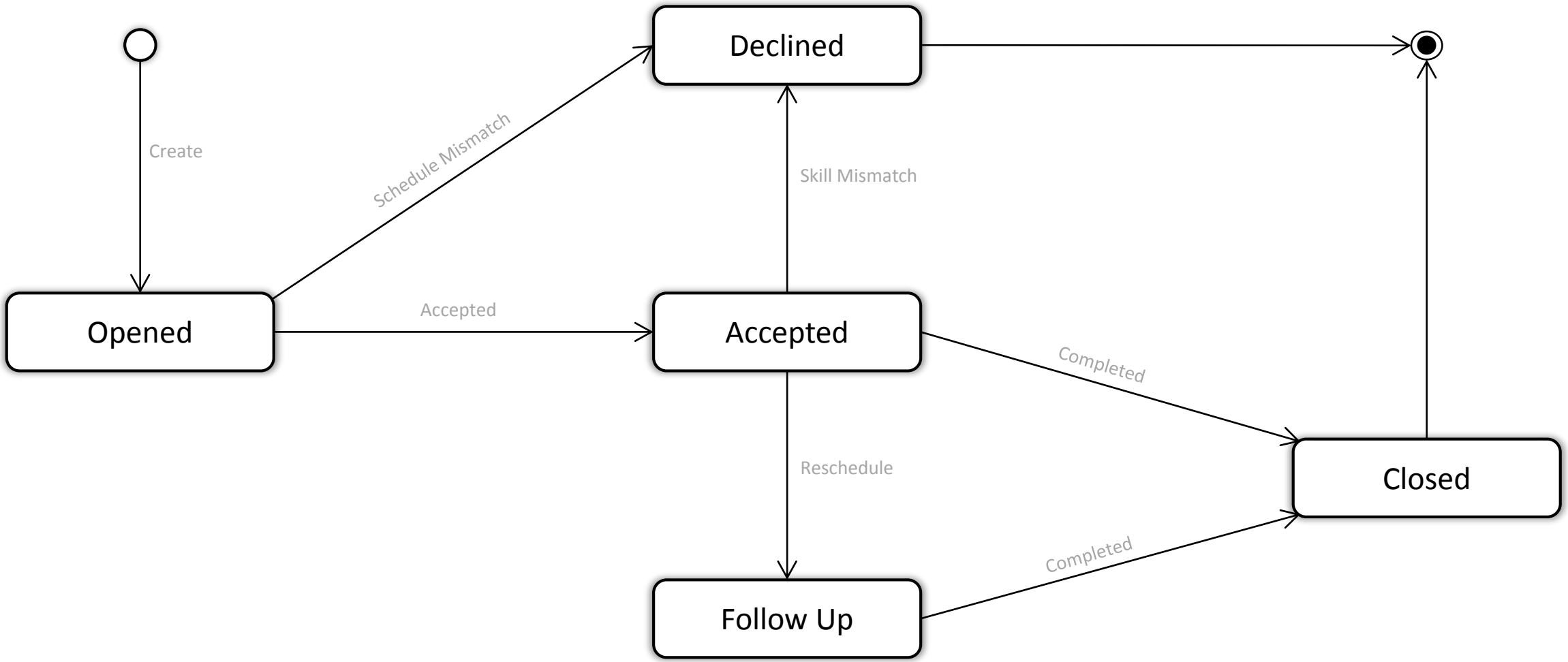
Service Provider UI

- Home Screen
- Add New Service
- List Service Requests
- View Request Details
  - Call or SMS
  - Accept
  - Update Status
  - Mark for follow-up
  - Accept Payment
  - Provide Feedback

Consumer UI

- Home Screen
- List Services
- List Service Providers
  - Location Map
- Request Service
- List Service Requests
- View Request Details
  - Report Issues
  - Accept Resolution
  - Make Payment
  - Provide Feedback

Service Request Lifecycle – Activity Diagram



Non-Functional Requirements

- Support Multiple Devices & Form Factors
- Offline Access
- Data Synchronization
- Role Based Access Control
- Data Security in Storage & Transit
- Task Efficient and User Friendly
- Minimize Network Traffic
- Reliability and Fault Tolerance
- Maintainability
- Crash Reporting
- Usability
- I18N, L11N – expansion to other geographies in future

Non-Functional Requirements [Continued]

- Data Archival
  - Once in 2 years
- Total Number of Users
  - 50000 (10% increase per year)
- Number of Concurrent Users
  - 200 (10% increase per year)
- Response Times
  - Home Screens & Forms: 5 seconds
  - Reports: 10 seconds
- SLA
  - Tier-2 Application SLA

App Development Technology

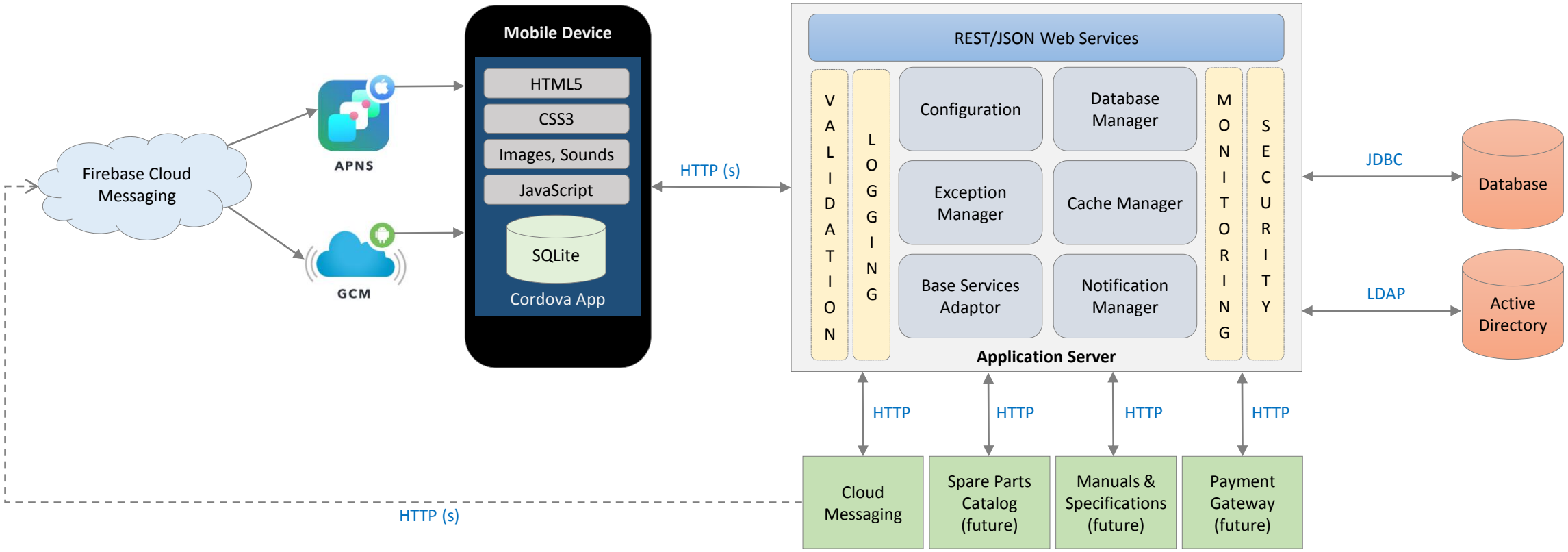
- Ionic & Cordova
  - App needs to work on at least 2 platforms, Android & iOS
  - App needs to go live in a limited time & budget
  - App has simple user interfaces, and easy on hardware resources
  - Staffing, maintenance and support will be easier, and cost-efficient
  - Rich UI libraries and plugins
  - Active developer community
  - Free and Open Source

Complete Technology Stack

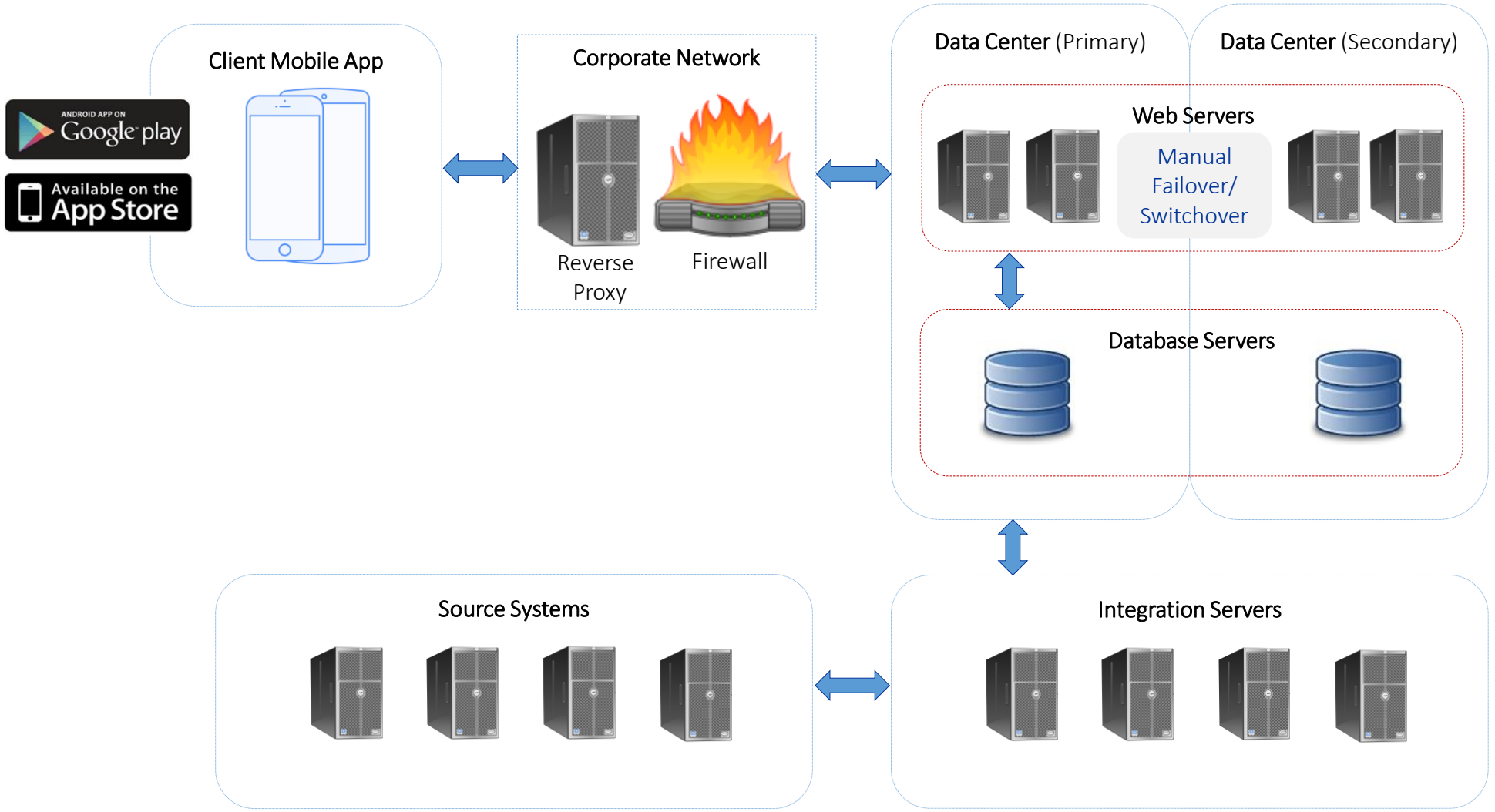
Server Side Component	Technology
Web Container	JBoss EAP 6.4
Persistence	Spring JPA
Database	Oracle 12c
Caching	Redis
Logging	Log4J
EAI	TIBCO
Continuous Integration	Jenkins
Business Services	Spring MVC - REST/JSON
Notifications	Firebase Cloud Messaging

Client Side Component	Technology
Static Content	HTML5 and CSS3/SASS
Client Side MVC	Ionic & AngularJS
Responsive Web Design	Bootstrap
Device Capabilities	Apache Cordova
JSLint	Static code quality checks for JavaScript
Testing	Karma & Jasmine
Offline Storage & Cache	SQLite & LocalStorage
Beta Testing	HockeyApp
Crash Reporting	Crashlytics
Remote Testing	Perfecto Mobile
Usage Analytics	Firebase Analytics

Application Architecture



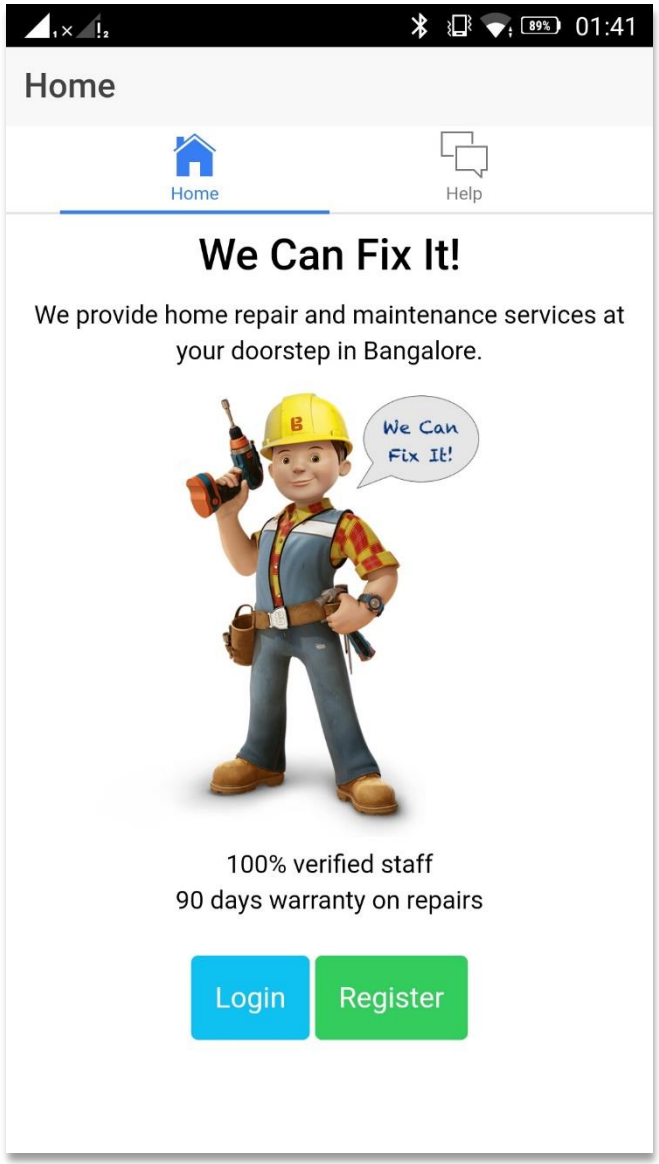
Deployment Architecture



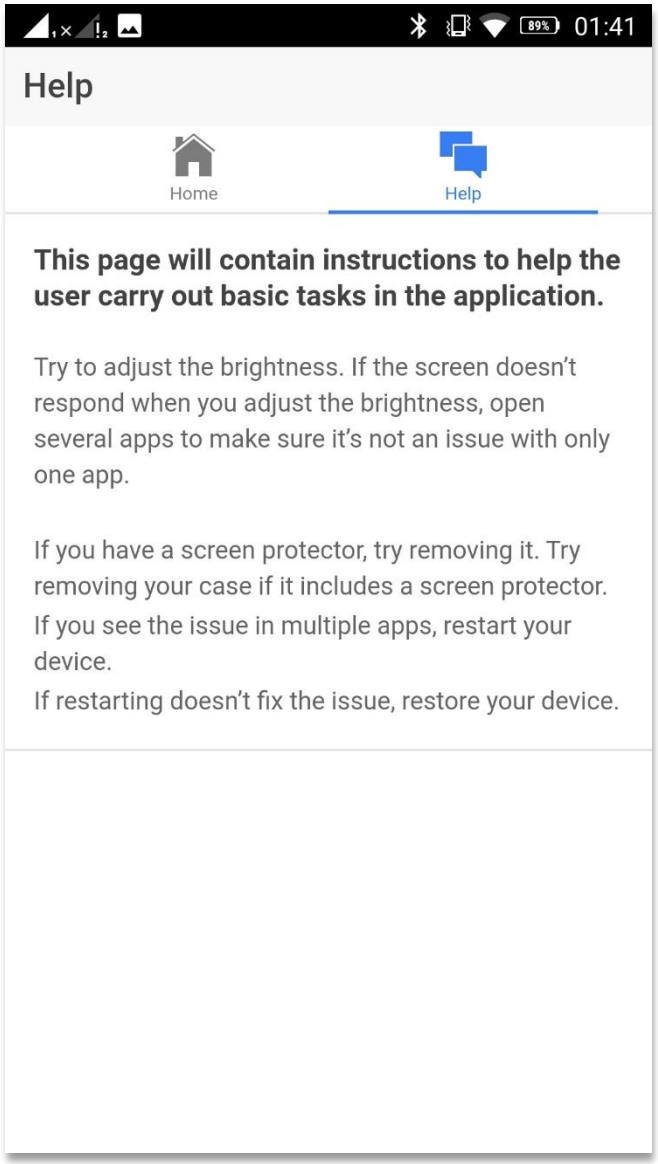
Capacity Planning

Environment	Server Specs	CPU / RAM	Disk Space	Type	Server Type
Production (Identical setup in 2 Data Centres)	Linux Server (2)	4 Core / 32 GB	50 GB	JBoss 6.4 EAP & Redis	Virtual
	Database (1)	4 Core / 16 GB	200 GB	Oracle 12c	Virtual
Staging	Linux – JBoss (2)	2 Core / 32 GB	50 GB	JBoss 6.4 EAP & Redis	Virtual
	Database (1)	2 Core / 16 GB	200 GB	Oracle 12c	Virtual
Testing	Linux – JBoss (2)	2 Core / 32 GB	50 GB	JBoss 6.4 EAP & Redis	Virtual
	Database (1)	2 Core / 16 GB	200 GB	Oracle 12c	Virtual
Development	Linux – JBoss (2)	2 Core / 32 GB	50 GB	JBoss 6.4 EAP & Redis	Virtual
	Database (1)	2 Core / 16 GB	200 GB	Oracle 12c	Virtual

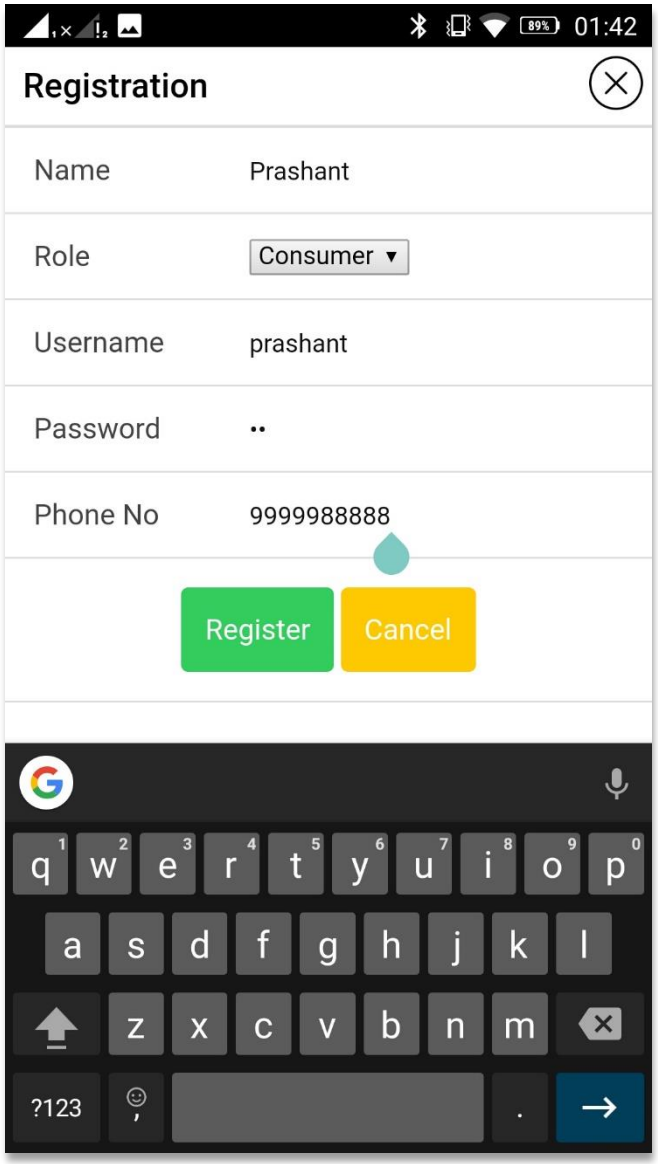
App Screenshots



Home Screen

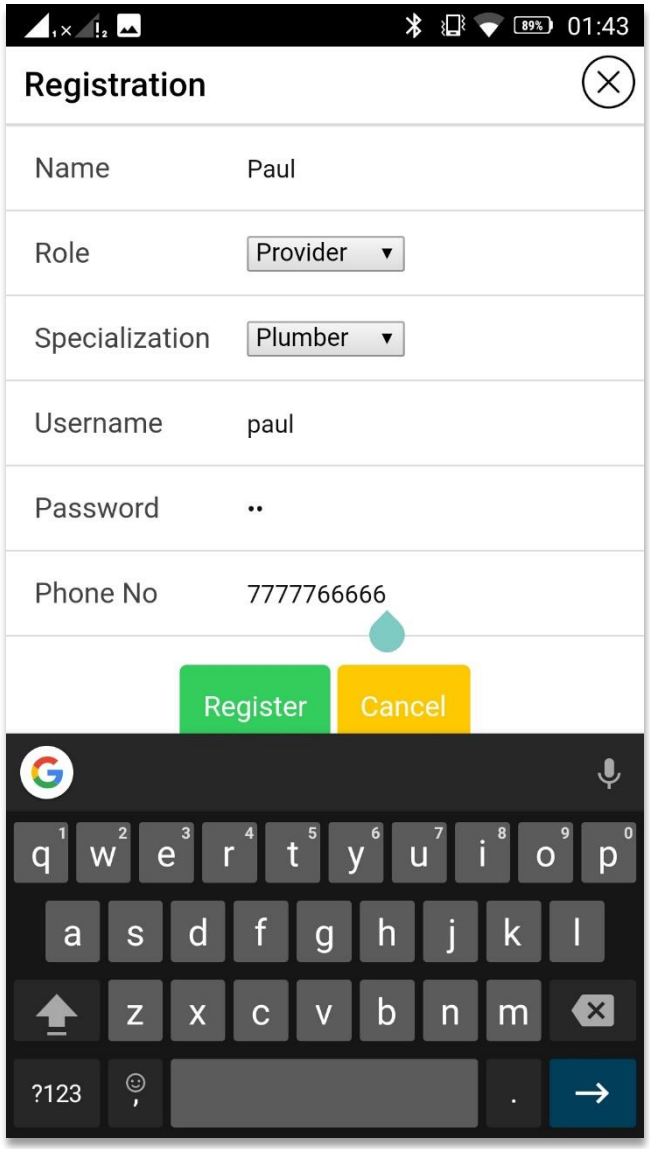


Help Screen

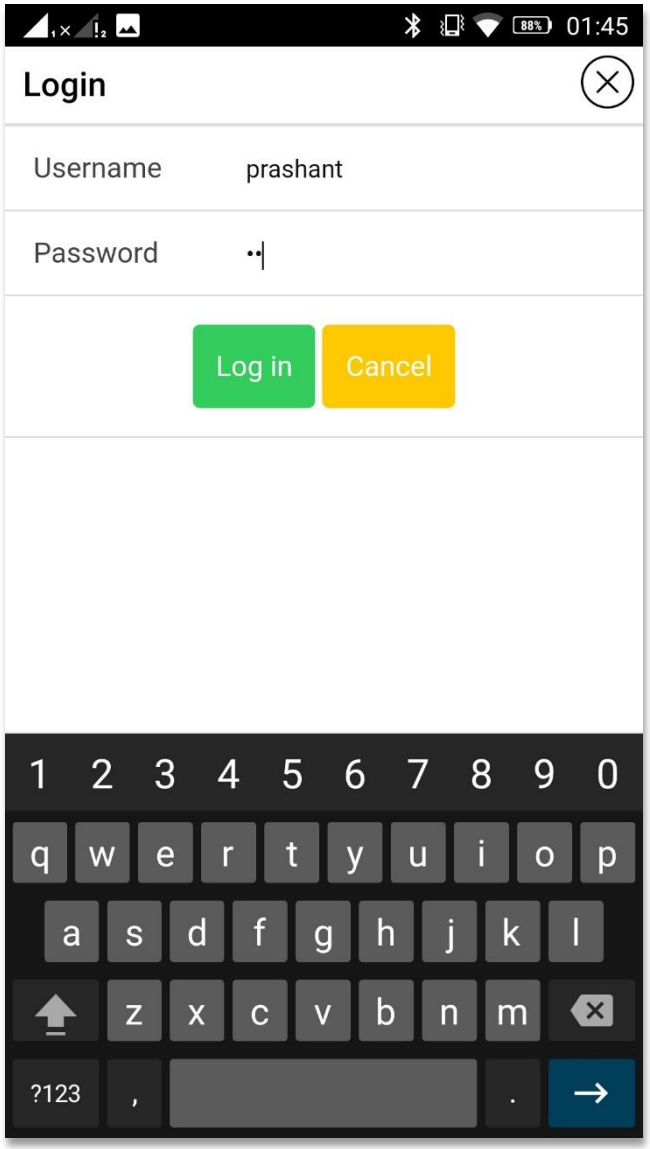


Consumer registers

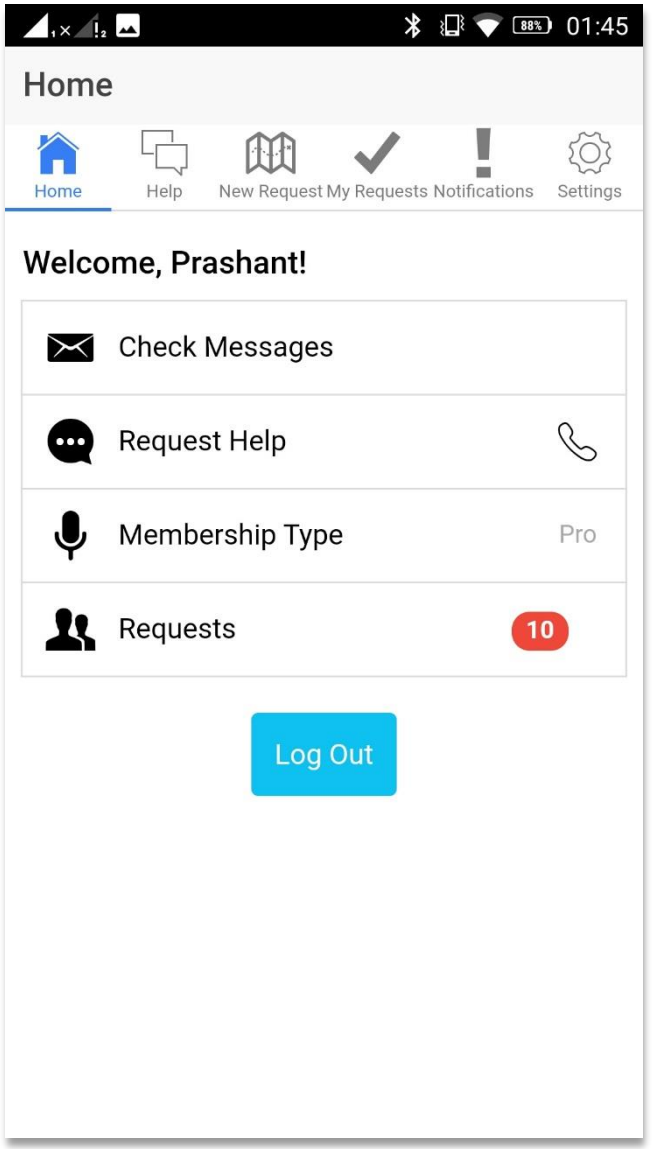
App Screenshots



Provider registers as a Plumber

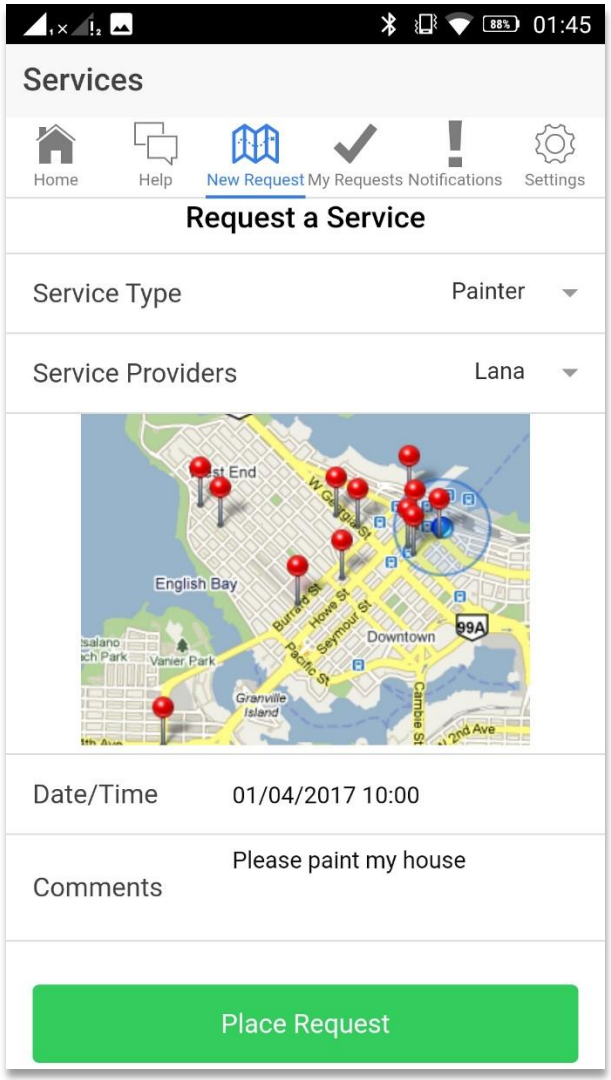


Consumer logs into the App

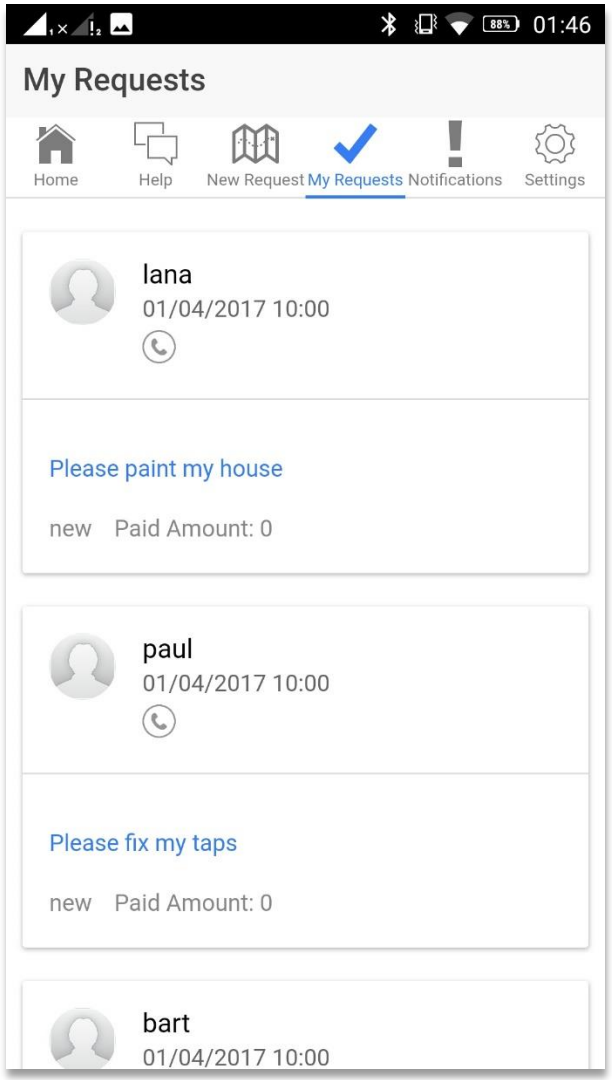


Consumer's Home Screen

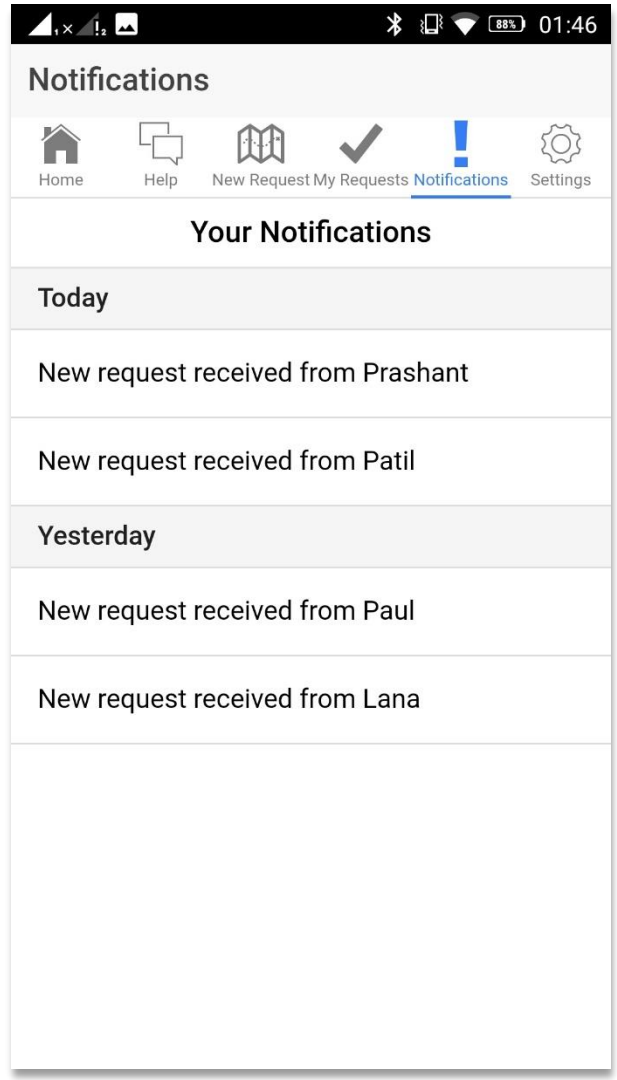
App Screenshots



Consumer requests new service

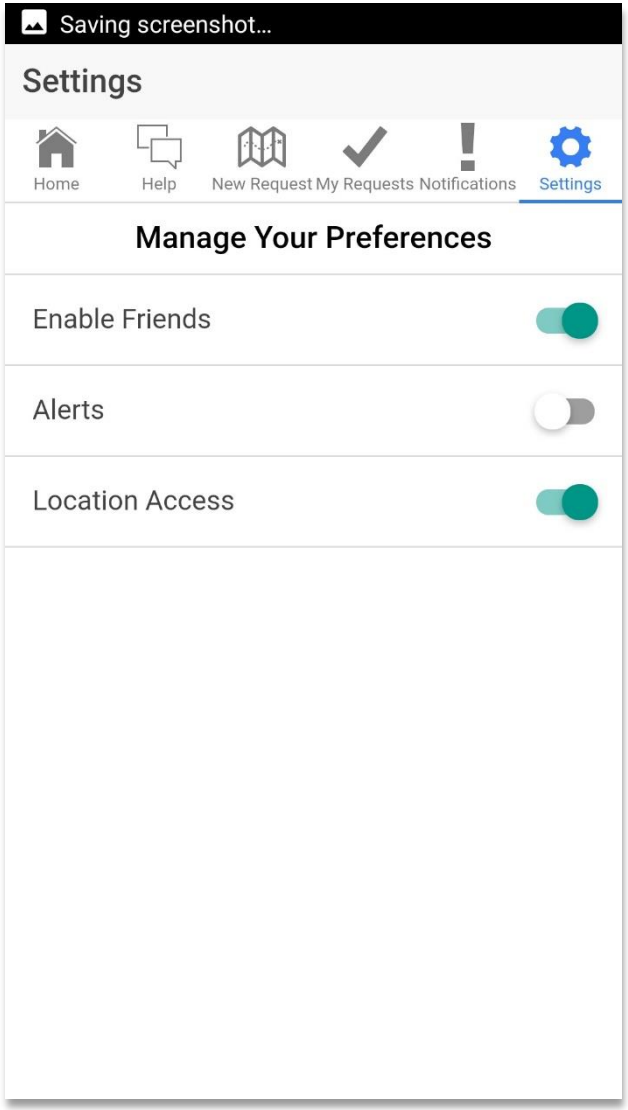


Consumer’s list of service requests

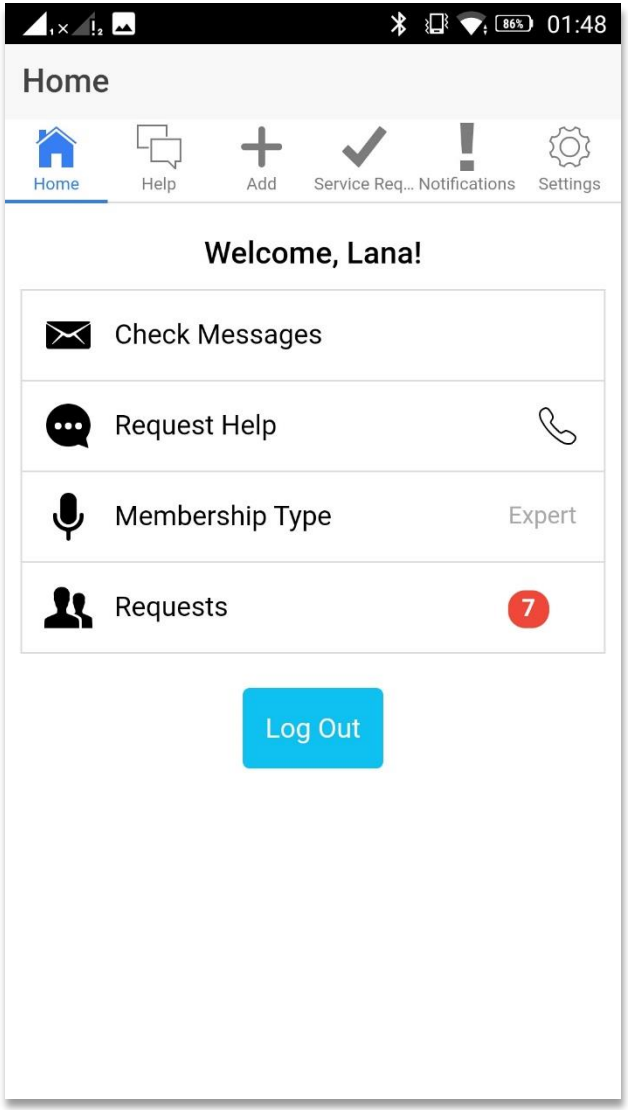


Consumer’s notifications

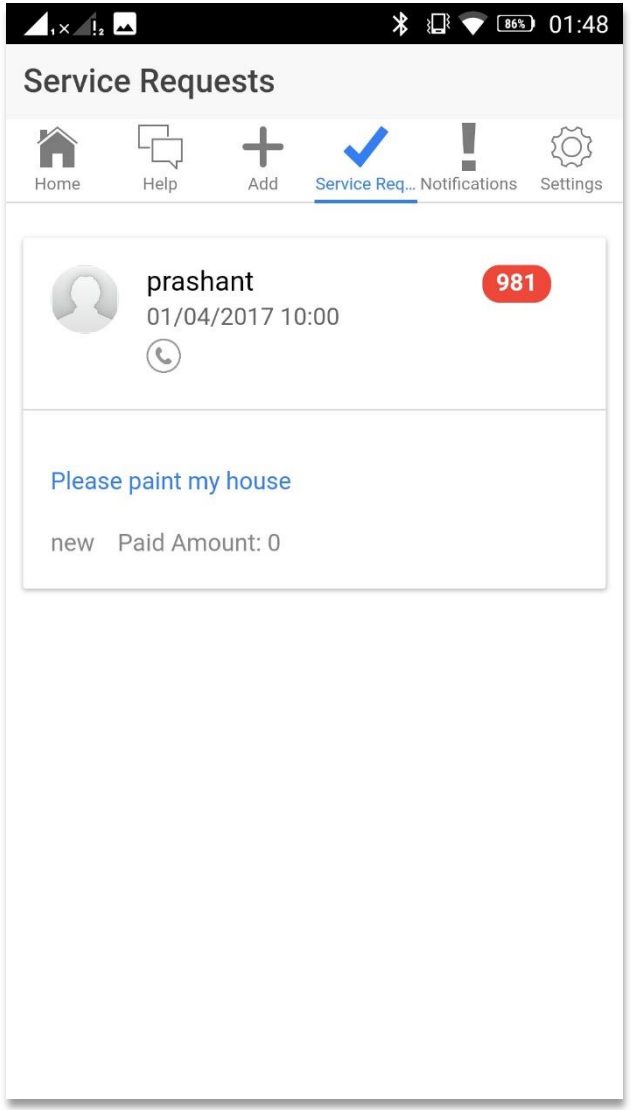
App Screenshots



Consumer's App Settings

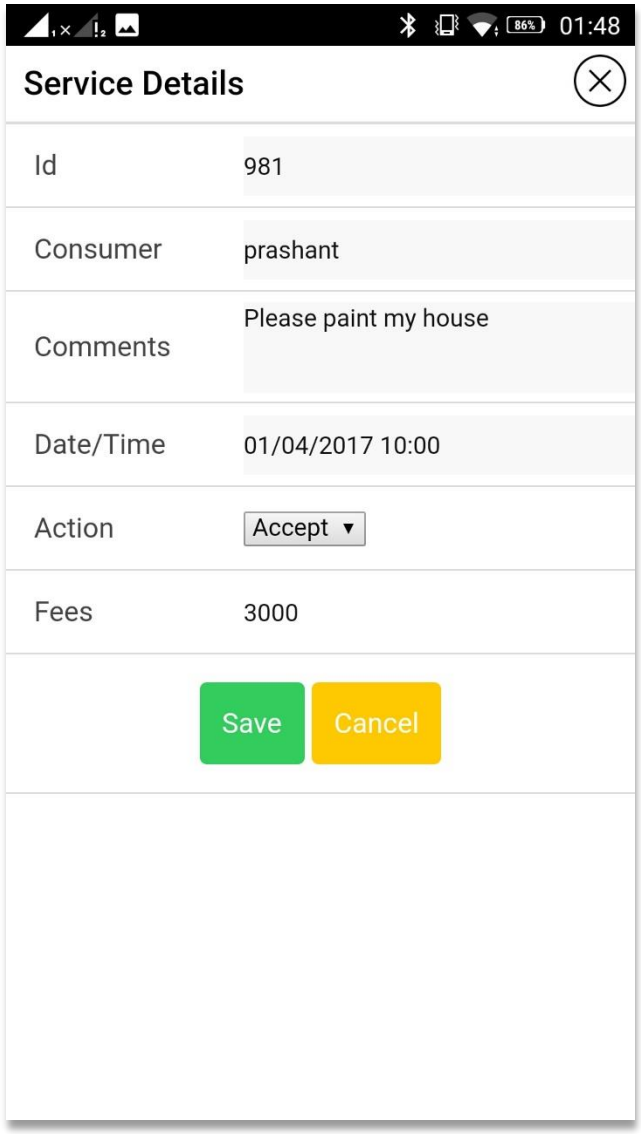


Provider's Home Screen

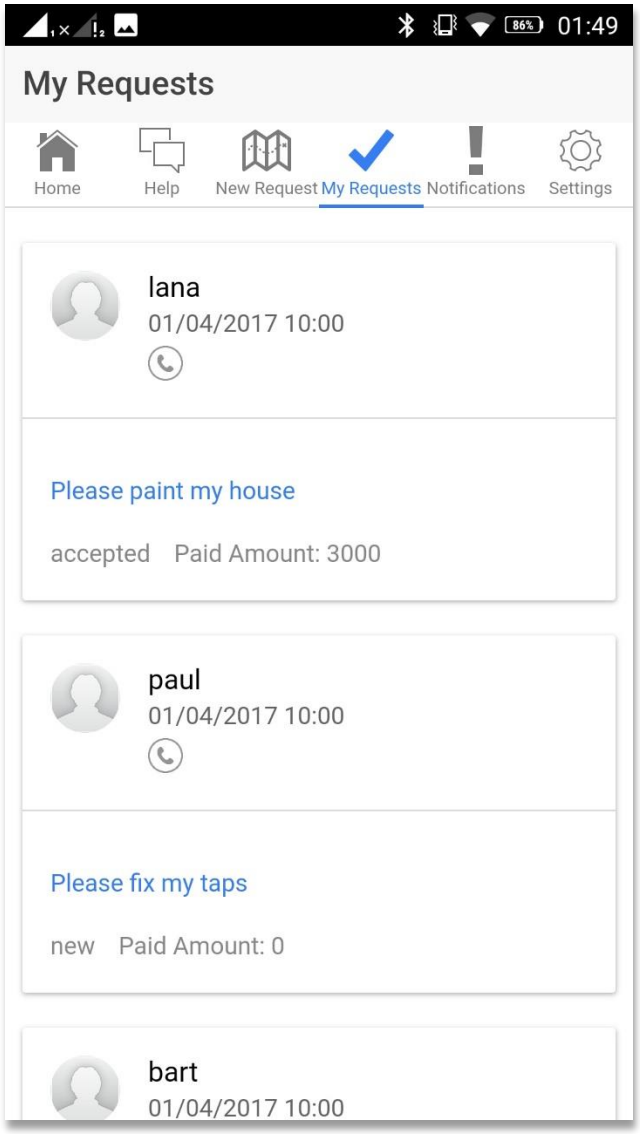


Provider's Task List

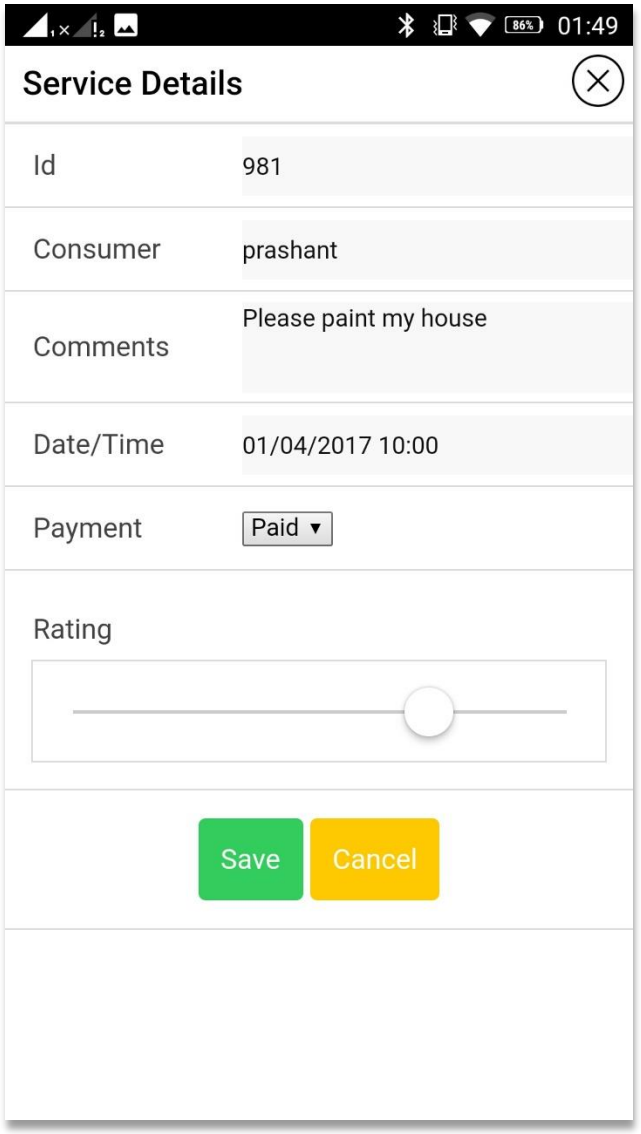
App Screenshots



Provider Accepts Request

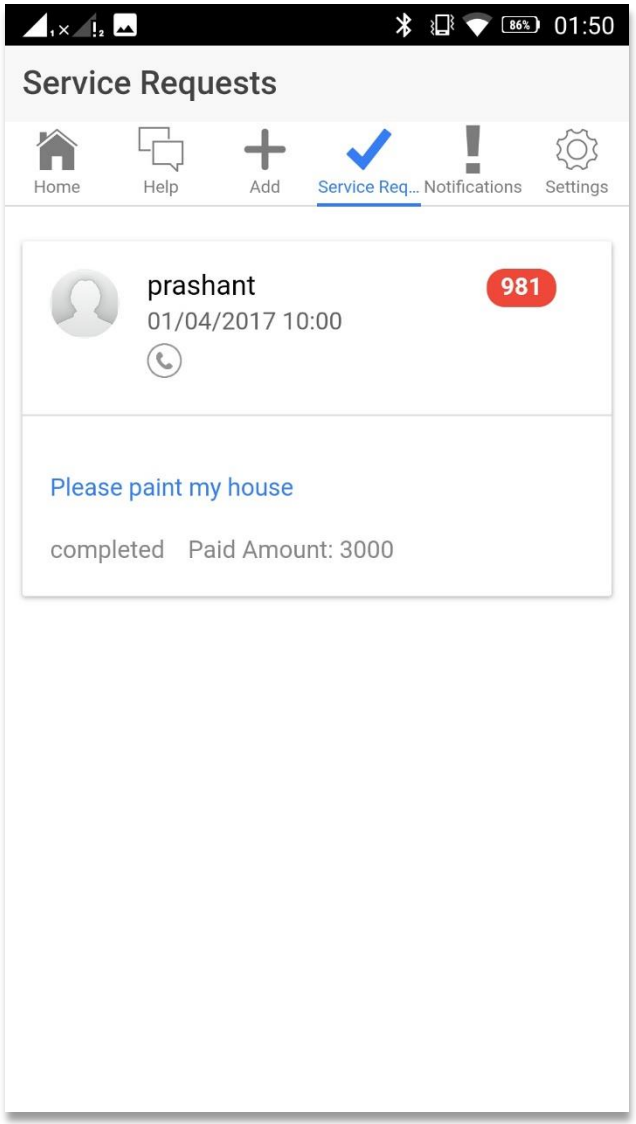


Consumer views Accepted Request

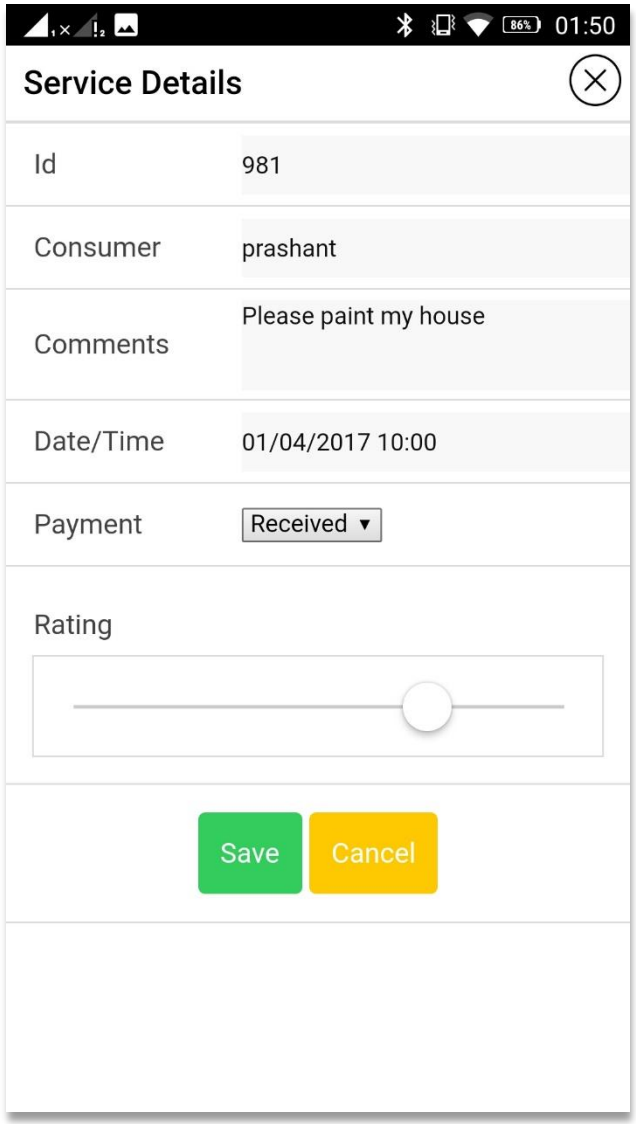


Consumer Closes Request

App Screenshots



Provider Views Request Update



Provider Closes Request