
GitHub Username: goinskj

Delivery Peer Review

Description

Delivery Peer Review is an auditing tool that gives courier drivers the opportunity to capture data directly related to the performance of their delivery. Customers can use their mobile device to complete a form submission review of the couriers delivery performance based on customer service and compliance related items. This will help governing agencies make better laws for its citizens and help improve the overall on-demand economy based on data collected from the peer review submissions.

Intended User

Intended users include delivery drivers, delivery managers, licensed delivery companies, compliance agencies, and city governing agencies in the cannabis industry (and their respective customers).

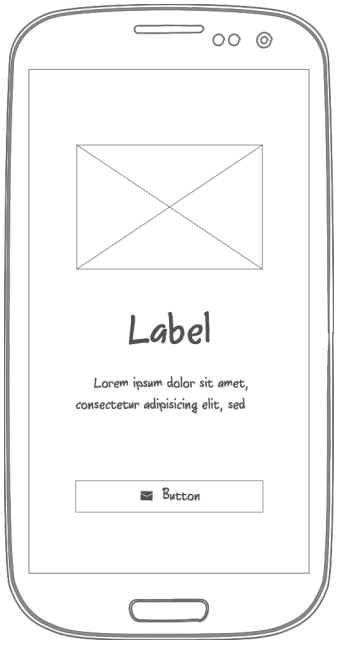
Features

- Saves information
- Reporting
- Rating system
- Real type data retrieval

User Interface Mocks

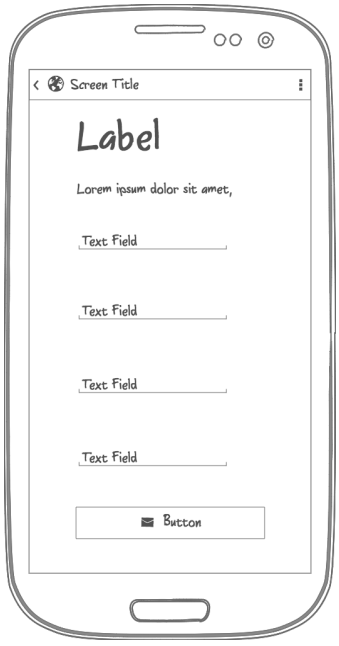
The below mocks were created using www.ninjamock.com.

Screen 1



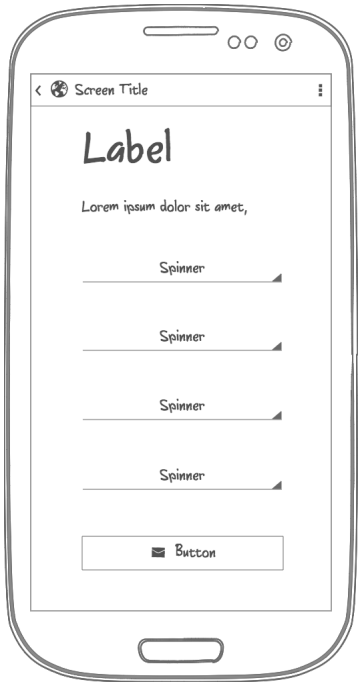
Start screen that prompts user to start a new peer review.

Screen 2



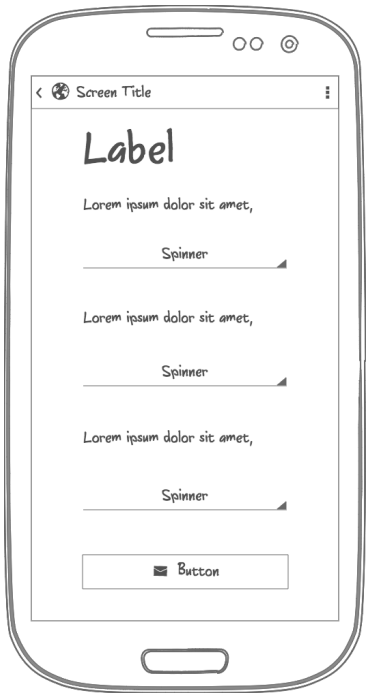
Activity that collects user information necessary for review report.

Screen 3



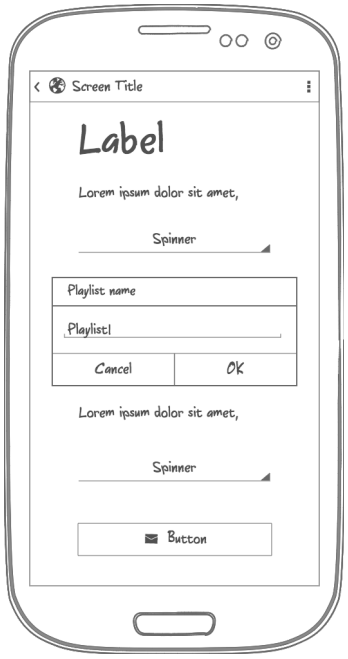
Activity that collects driver information necessary for matching POS transaction to report.

Screen 4



Activity that prompts the rater with questions regarding courier's performance during delivery.

Screen 5

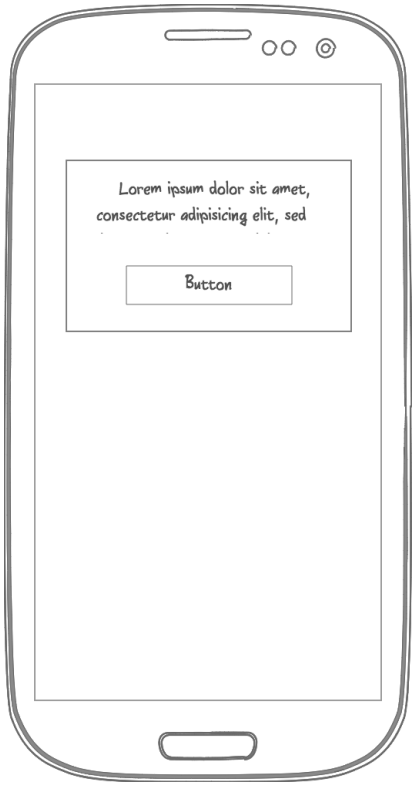


Alert dialogue that presents user opportunity to verify their selections. If confirmed, results are sent to database.

Screen 6

Activity finishes session and return user to start screen.

Screen 7 - Home screen widget



Home screen widget. Prompt user to continue with review.

Key Considerations

How will your app handle data persistence?

The app will use Room Database for data persistence.

Describe any edge or corner cases in the UX.

Users can toggle back and forth between pages 1 - 5 of the app, but once page 6 has been reached, the report has been sent and user must complete another review.

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

Picasso will be used to handle the loading and caching of images. Data binding library for eliminating unnecessary code, spinner and drop down libraries and material design library for

aesthetics, and firebase libraries for data retrieval. All versions of libraries that are used in stage 2 will be stable releases.

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

Google locations will be used to verify users location and real time data base will be used to retrieve driver information.

Next Steps: Required Tasks

This is the section where you can take the main features of your app (declared above) and break them down into tangible technical tasks that you can complete one at a time until you have a finished app.

Task 1: Project Setup

Overview Steps

- Configure libraries and dependencies
- Setup values folder with necessary values
- Setup layout files for pages 1 - 3
- Generate java code for pages 1 - 3
- Setup layout files for pages 4 - 6
- Generate java code for pages 4-6

Task 2: Implement UI for Each Activity and Fragment

- Build UI for pages 1 - 3
- Build UI for pages 4 - 6

Task 3: Generate Java Code

- Generate Java code for pages 1-3
- Generate Java code for pages 4-6

Task 4: Configure Libraries & Dependencies

- UI related libraries
- Data persistence libraries
- Google location libraries
- Database libraries

Task 5: Setup Values Folder

- Strings
- Layout
- Dimensions
- Colors
- Styles

Task 6: Tablet Responsiveness

- Layout for tablet
- Java code to accommodate tablet layouts

Task7: Finishing Touches

- Generate code for tests and run testing
- Clean and rebuild project
- Share project to Github account
