



APRIL 2, 2022

MOTORBIKE RIDE SYNCHRONIZATION

A PROJECT REPORT ON PROTOTYPING AND USER TESTING

SAHIL SHAHEEN, SANDHEEP GOPINATH
FOR HUMAN-COMPUTER INTERACTION, PLAKSHA TLP '22
Instructor: Sudheendra Hangal
TA: Subhash Kashyap




Table of Contents

<u>EXECUTIVE SUMMARY</u>	<u>2</u>
<u>INTRODUCTION</u>	<u>2</u>
<u>PROBLEM STATEMENT</u>	<u>2</u>
<u>PROTOTYPING</u>	<u>2</u>
OVERVIEW OF THE PROTOTYPE	3
<u>USER TESTING</u>	<u>5</u>
HYPOTHESES.....	5
METHODOLOGY.....	5
FOR THE APP.....	5
FOR THE CRASH ALERT FEATURE	6
SUMMARY.....	7
<u>LEARNINGS</u>	<u>7</u>
SELECTING LOCATIONS.....	8
CONFUSION AROUND “PAUSE TRACKING” FEATURE	8
OTHER LEARNINGS/SUGGESTION.....	9
<u>CONCLUSION</u>	<u>9</u>

Executive Summary

This report follows the prototyping and user testing of a motorbike ride synchronisation app. It covers an **introduction** to the landscape of motorbiking within the country, the **problem statements** addressed by the app, **existing work** with regards to the problem, an **overview of the prototype** and the process of building it, **methodology** for user testing, and **findings** from the user tests conducted. The report concludes with our learnings from the process *divorced from the app itself*.

Introduction

The motorbike industry in India has been growing rapidly over the last few years from niche to a thriving industry. This has led to many people recognizing not only the utility of motorbikes but bringing out the passion in them and bonding with likeminded people. Many riding clubs have been formed all over India, each of them conducting multiples rides in a year. The riders face many problems during group rides, which we evaluated as a part of the Product Management course. We conducted multiple user interviews and we have produced a solution which helps in resolving the major problems faced by riders which is discussed in detail below.

Problem Statement

Through personal experience with the activity and interaction with the community, we were able to identify that there is no effortless way for motorbike riders to track fellow riders while they are going on rides in groups. The current solution for this problem varies across clubs. Some of them use checkpoints to make sure everyone is on track. These checkpoints are often 50-100km apart. Riders also make use of different platforms like Google Maps, WhatsApp location services, group chats, phone calls, etc. to keep track of fellow riders. We also learnt that accidents and breakdowns which happen during the ride often take a long time to reach others in the group.

Prototyping

The background work for the project was done as a part of Product Management course, where we did user interviews with multiple riders and clubs to understand the problem. Once the problem was identified, we took reference of similar platforms, ESR (Eat Sleep Ride), in the market. We evaluated the platform to understand what could be improved and we used it as the base for building our interface.

We found that the motorbike riding community is truly diverse with people from different age groups and backgrounds, and hence we identified the need for a simple app design. Most of the user's interaction with the app happens before the ride, and during the ride the

phone would often be placed on the handlebar. We had to design the interface in such a way that the user has least interactions during the ride to not distract them.

Overview of the Prototype

We produced a [Balsamiq prototype](#) with the following features:

- Creating a ride
- Joining a ride
- Tracking fellow riders
- Crash alert

The user gets to see an overview of the rides in progress, the completed and upcoming rides from the rides screen.

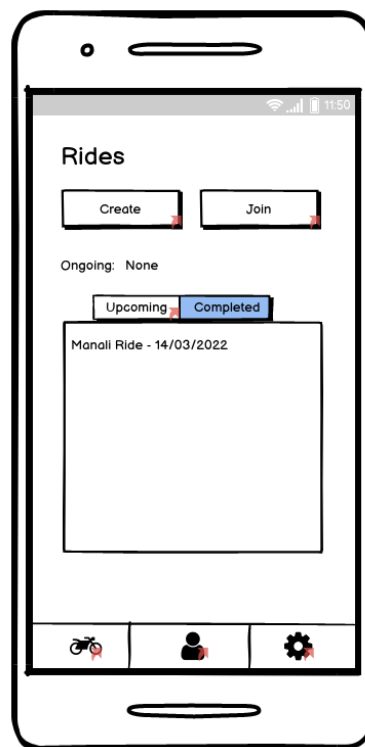


Figure 1: Rides page

The user will be taken to the below screen as they click on the “create” button. During this process, the user will be asked to enter the basic ride details and the locations where the ride starts and ends. The user is also given an option to keep the ride open to public or private, and the option to add checkpoints so that they can plan the stops in between.

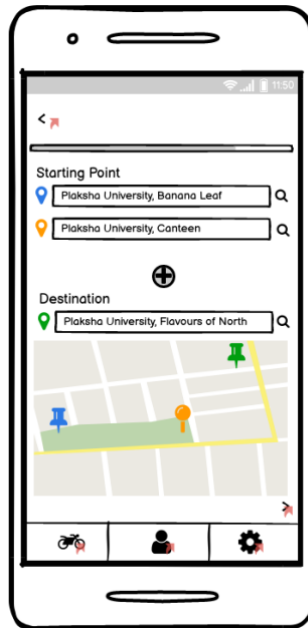


Figure 2: Screen from create ride workflow

Once the ride is created, users can join a ride by entering the ride ID. The user will get a tracking page upon entering, where they will be able to track fellow riders.

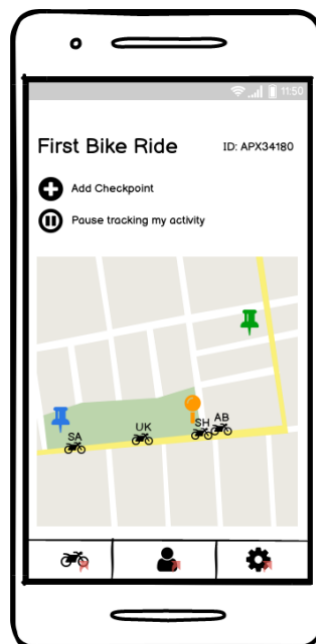


Figure 3: Tracking page

If any of the riders encounter a crash/accident, the user will be alerted with the screen flashing at a certain frequency. When the user responds to this, they are taken to the SOS page where they will have option to contact the user who encountered the event, or the rider closest to them.



Figure 4: Crash alert page (after the user responds)

User Testing

Hypotheses

After preparing the prototype, we had three main hypotheses we wanted to test.

- 1) **Users will be able to intuitively understand the app capabilities and interface:** We had designed the app to be as minimal as possible. However, we recognised that there is a trade-off that exists with interpretability (without tutorials or extensive documentation). We wanted to know if we were able to find the sweet spot with our initial prototype.
- 2) **Users will be able to create/join a ride without assistance:** At the core of the application is the ability to create/join a ride. We wanted to make sure that the user could go through the form to achieve these tasks without taking assistance and understand the underlying mechanism (ride IDs).
- 3) **Users will be able to understand terminologies within the app:** During the prototyping phase, we had deliberated on what terms to use to refer to certain features and settings. We wanted to see whether the terms we settled on would convey the meaning effectively.

As for the crash alert feature, we wanted to test the hypothesis that the user would quickly respond to the flashing of lights designed to attract their attention.

Methodology

For the app

The user test methodology consisted of the typical flow of briefing the user, assigning tasks, evaluating the user's performance, and summarizing and incorporating learnings. But to test

the specific hypotheses with regards to our prototype, we came up with the following methods.

- **Do not brief the user about the specific capabilities of the app:** In the briefing, we only mentioned that the app caters to the problems of motorbike riders that travel in groups. This helps us test how interpretable the app and its features are.
- **Prompt the user to talk through certain terminologies:** Throughout the test, we asked the user to tell us what they thought certain terminologies referred to. Examples include terms like “ride ID”, “pause tracking”, etc.

User Briefing Script

*Hello, __. Thanks for agreeing for this user test. You are going to be testing the interface of an app. Broadly speaking, this app is aimed at motorbike riders who ride in groups. As for the features, that is something we want you to discover by yourself. The goals of the test would be to test the intuitiveness of the interface, discoverability of the features like previously mentioned and ease of accomplishing certain tasks that we will be assigning to you. This test should take approximately 15 minutes. It is important for you to know that **we are testing the app, not you**, so making mistakes is perfectly okay. In fact, the more mistake you make, the more learnings we have from this test. If you are confused about how to do something or if you are not sure what a particular element/information means, then **dwel on it for as long as you can**, but we will chime in eventually in case you are not able to move ahead. We encourage you to **think out loud** as you see the various screens of the app and try to take some time with each screen before you move on from it or interact with any of the elements. If you want to take a break or are unable to finish the test, that is perfectly fine. With your **permission**, we will be recording the screen and audio from the user test and with additional permission, we would like to present parts of it for a presentation. ... Let us get started.*

For the crash alert feature

To test the crash alert feature, we conducted a Wizard of Oz test. The bike ride was simulated by pushing the *rider* seated on an office chair with wheels for a period of ~30 seconds. The user was asked to hold a phone to mimic as best as possible a phone placed on the handlebars of a motorbike. The phone played a video that was a screen recording of the navigation screen of the Apple Maps app for ~20 seconds and then a strobe light application that flashed a bright white screen at a regular frequency. The user was asked to *ride safe* and keep their *eyes on the road* and was also told that they could look at the screen time from time to time to check Maps.

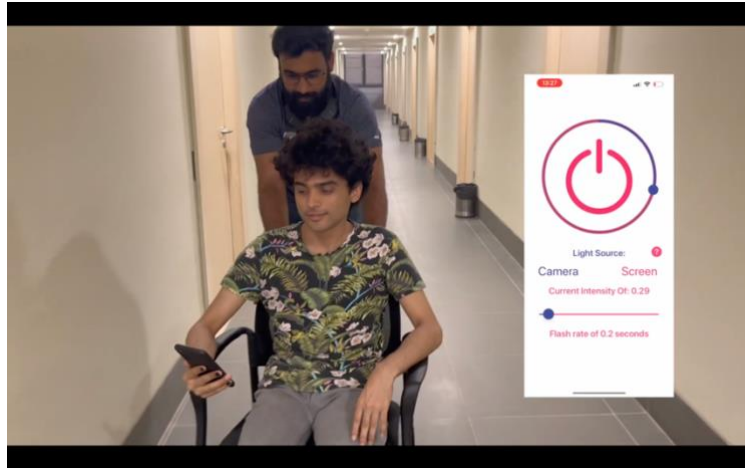


Figure 5: Screenshot from the [crash alert test](#)

Summary

We conducted four user tests for the app and two user tests for the crash alert feature. Below are the results we obtained for the app:

- Users **could** intuitively understand the capabilities of the app and the interface.
- Users **could** create/join rides without assistance
- Users **could not** understand certain terminologies within the app

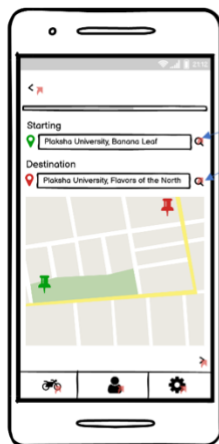
As for the crash alert feature, both the users responded to the flashing screen within 2-3 seconds. However, given the less-than-ideal modelling of the actual scenario, more refined tests should be conducted to incorporate:

- Better modelling of the user's attention during a ride
- Phone positioning
- Conditions of the ride (For example: what happens in conditions of strong sunlight?)

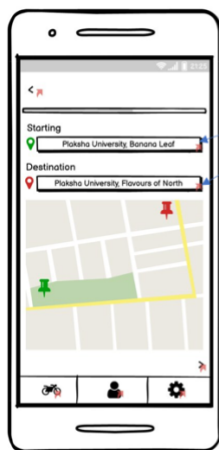
Learnings

There was a lot of learnings with regards to the Balsamiq prototype and suggestions as well. Below we list the main ones and show how we incorporated them into the prototype.

Selecting locations



Users were finding it difficult to click this button. They were clicking on the text box as that is how it works usually.



We replaced the icons with responsive textboxes making it easier for users to interact with the page.

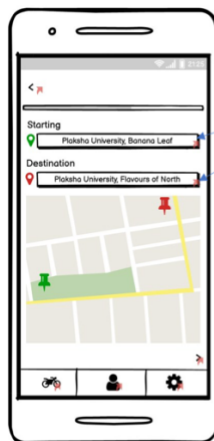
In the initial prototype, we designed the page to select locations (in the create ride workflow) in such a way that the user had to click on a button to go to another page where they would fill in the location. This was rectified as shown above.

Confusion around “pause tracking” feature

In all the user tests, the pause tracking feature, which was meant to stop tracking ride metrics (distance, time, etc.), was not understood by the user or taken to be a different feature (stop location sharing), or like in the figure, users could not understand the need for it.



Users were not able to understand the need for this, although we thought this would be an important feature in terms of user control and freedom



We replaced the icons with responsive textboxes making it easier for users to interact with the page.

Other learnings/suggestions

- User's marker on the tracking page should be highlighted for quick recognition
- There should be a home page with quick links and statistics about the user's rides
- Control should be given to the user to start the ride, instead of automatically starting the ride based on time/location
- There should be quick action buttons to alert other riders of certain scenarios like when the user is out of fuel, or the vehicle requires maintenance

Conclusion

Through the course of the project, we were able to put into practise our learnings about prototyping and user testing. We enjoyed working with Balsamiq which helped us create and iterate on our prototype in a fast manner. We realised that even if we have a good understanding of the principles of usability, translating that into a product is a hard job. We also realised the value of conducting user tests in a systematic manner – recognising the importance of components such as pilot testing, scripts, and structured notetaking.

Finally, we would like to thank the instructor Prof. Sudheendra Hangal and the TA Subhash Kashyap for a thoroughly illuminating and enjoyable course and all the guidance they provided through it.