# 

# Safe Riders Web Application

Created by the Riders[[1]](#footnote-1)

## /Users/Conor/Desktop/Screen Shot 2016-04-21 at 1.57.25 PM.png

## Table of Contents

Safe Riders Web Application 1

[Table of Contents](#_Table_of_Contents) 2

1 Riders Overview 3

[1.0 Team Overview](#_1.0_Team_Overview) 3

1.1  [Mission Statement](#_Mission_Statement) 3

2 System Overview 4

2.0: [High-Level User Overview](#_2.0:_High-Level_User) 4

2.1: [Expanded High-Level Overview (For Developers)](#_2.1:_Expanded_High-Level) 5

2.2: [Architecture Diagram](#_2.2:_Architecture_Diagram) 6

2.3: [Architecture Expanded](#_2.3:_Architecture_Expanded) 6

2.4: [Future Improvements](#_2.4:_Future_Improvements) 6

3 User Manual 7

3.0: [Student User Manual](#_3.0:_Student_User) 7

4 Planning & History 8

4.0: [Initial Weekly Milestones](#_4.0:_Initial_Weekly) 8

4.1: [Actual Progress](#_4.1:_Actual_Progress) 9

4.2: [Developer Logs](#_4.2:_Developer_Logs) 9

4.2.0: Conor’s Log 9

4.2.1: Zhibin’s Log 10

4.2.2: Jacob’s Log 10

# 1 Riders Overview

## 1.0 Team Overview

* ZHIBIN ZHANG (Head of Front-End Development)
* CONOR TRACEY (Head of Middle-End Development,

Documentation Editor)

* JACOB BROWN (Head of Back-End Development)

## Mission Statement

The mission statement of Safe Ride is “Safe Ride is devoted to the idea that a person regardless of gender identity, race, religion, or sexual preference should be able to go where they wish at night while feeling comfortable and safe.” To some extent, our mission statement is a firm “diddo”.

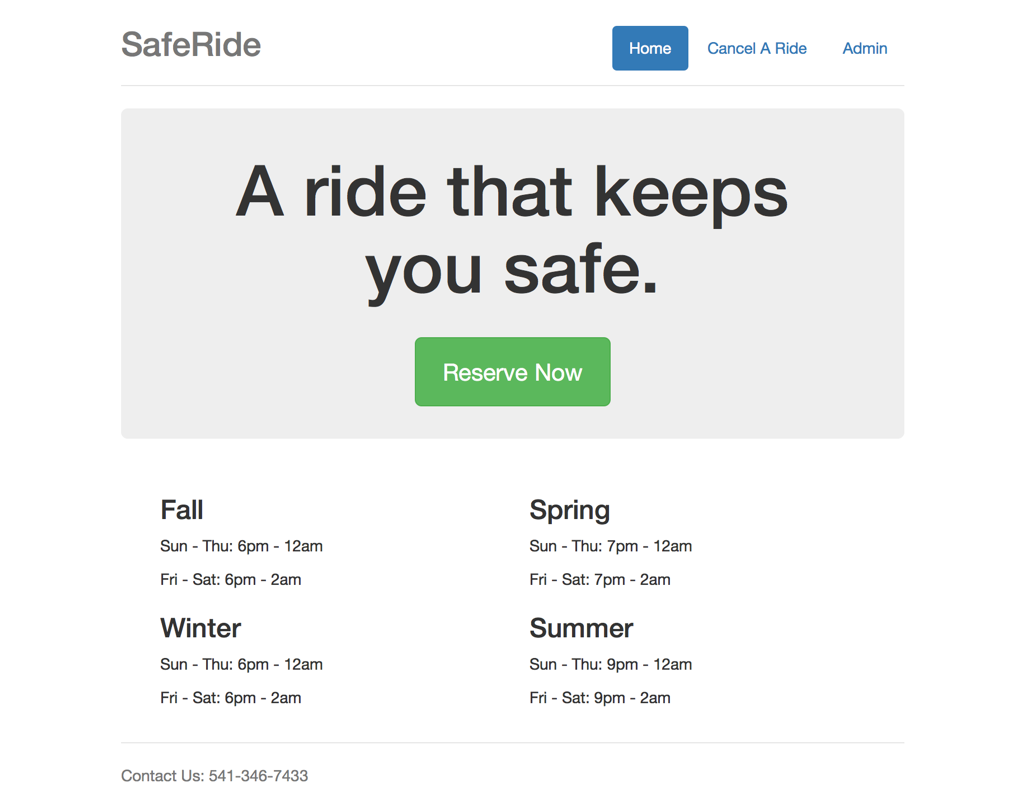
Our goal is to assist Safe Ride’s operation from both ends, making the work of Safe Ride staff easier, and making their service more accessible to the student community.

# 2 System Overview

## 2.0: High-Level User Overview

The purpose of this system is to allow student patrons of the Safe Ride service to send ride requests over the internet[[2]](#footnote-2), saving time for both students and Safe Ride staff.

To use the application, students connect to the host website and enter their information (name, phone #, pickup/ drop-off location, pickup time, and group size). This information is then cross-referenced with Safe Ride’s boundaries before being displayed to dispatch as a queue of request boxes with relevant information which they can copy-paste[[3]](#footnote-3) into their own, existing, spreadsheets.



The current version (1.0) of this application is supported on the Mozilla Firefox web browser. Accessing this application with Safari, Chrome, or on any mobile platform is not recommended at this time.

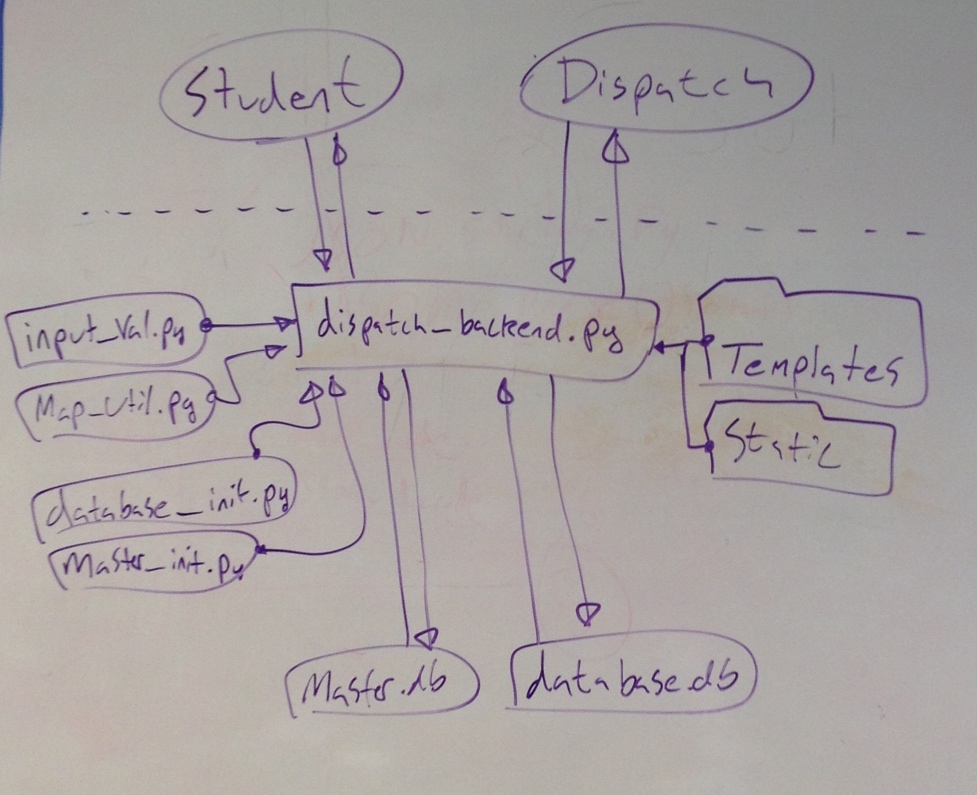
## 2.1: Expanded High-Level Overview (For Developers)

This system uses the Flask framework for python, as well as an open source Flask extension called Flask-GoogleMaps. It also incorporates a python extension called Geopy which allows geolocation, a vital part of input validation.

The system has two front-ends, one facing the student users, and the other facing the Safe Ride dispatch staff. Both of these are hosted on the same server with open access to student resources, and restricted access to dispatch resources.

Both front-ends are controlled through a middle-end application engine which runs on a server and handles all front-end requests with a single thread. The application engine has access to two back-end databases: one for the dispatch queue, and another for a “blacklist” for students who consistently misuse the service. The engine also has access to an input validation module to integrate with student POST requests to ensure that their pickup/ drop-off locations are within Safe Ride’s Boundaries, and their student IDs are valid[[4]](#footnote-4)

## 2.2: Architecture Diagram



## 2.3: Architecture Expanded

The main engine of the application in dispatch\_backend.py. This engine has access to the databases (and their initialization functions), the modules for input validation and google map creation, as well as the templates and static directories[[5]](#footnote-5).

The main engine formats and sends html templates, decides if input is valid, pulls and pushes data to the databases, and generally does all active work.

## 2.4: Future Improvements

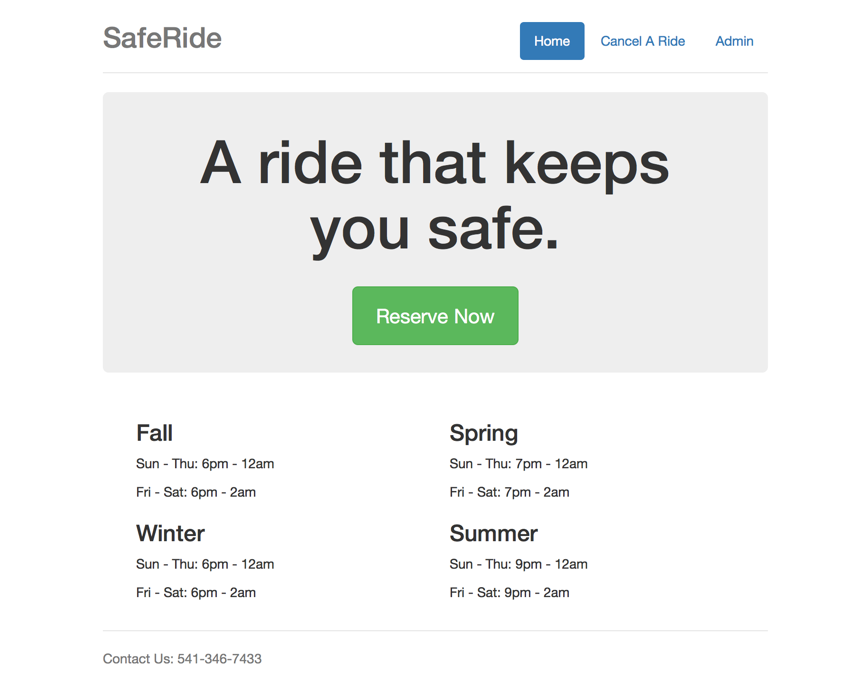
* Internet Security
* Add functionality to “Remove” button on “Cancel a Ride” Page

# 3 User Manual

## 3.0: Student User Manual

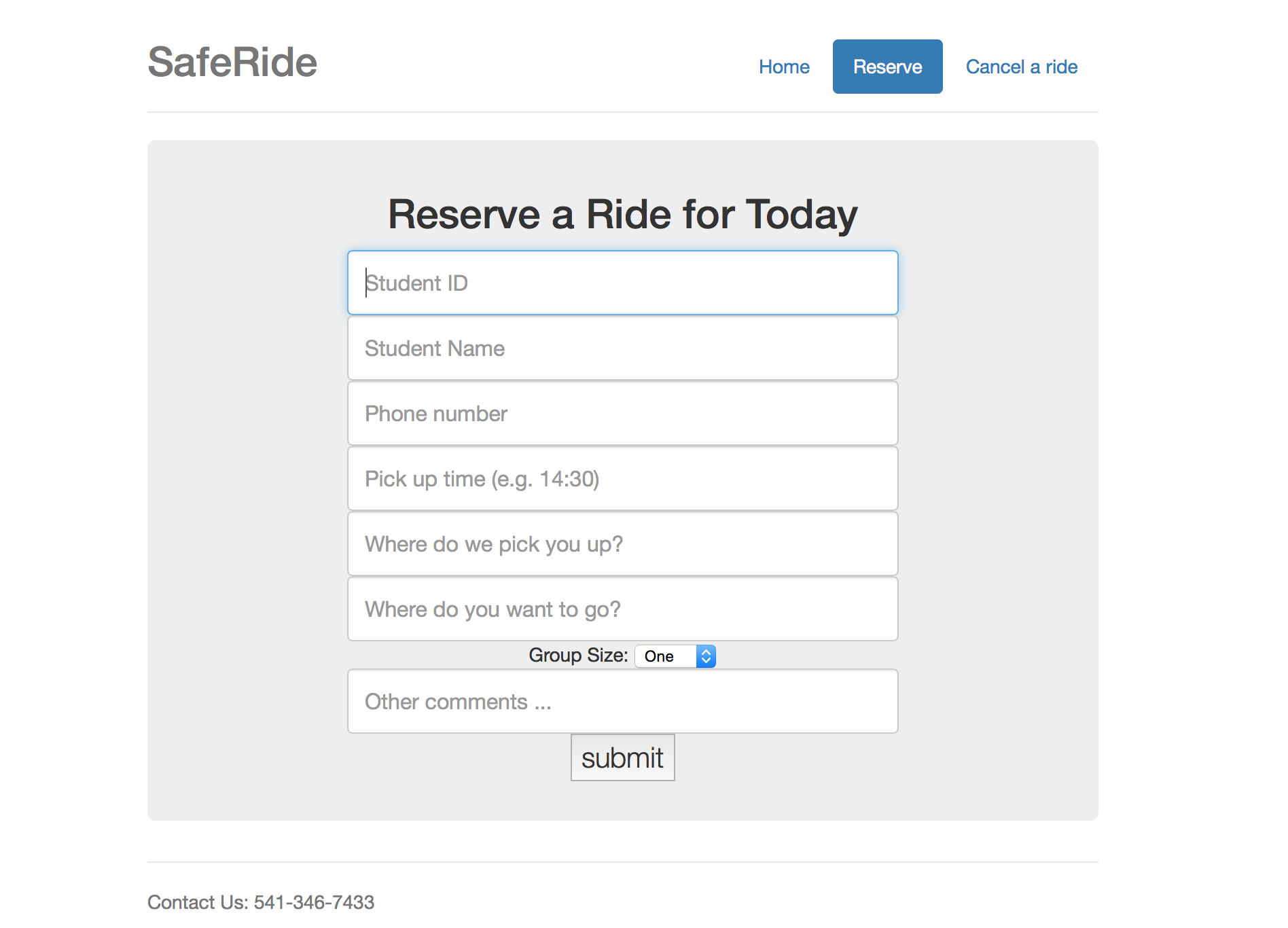
The Safe Riders application is designed to be relatively straight-forward for students to use.

1. The first page you will see when using this service as a student is the welcome page:



To reserve a ride as a student, the first step is to click the “Reserve Now” button in the middle of the page.

1. This will bring you to the reservation page. Simple enter all pertinent information, and hit the “submit” button.



# 4 Planning & History

## 4.0: Initial Weekly Milestones

Week 2:

* Build basic UI with all the proper text boxes that sends user to blank screen when they hit “Enter”
* A Google Map with Safe Ride boundaries outlined that lets you select where you are / where you want to be
* A basic database architecture that sorts by scheduled pickups and allows Dispatchers to verify / deny

Week 3:

* Add basic UI functions (i.e. “Safe Ride Successfully Hailed!” Page, “Error: Input not properly entered” page,“Safe Ride cannot accommodate you” page)
* Add functions to determine if a certain map location falls within the Safe Ride boundaries
* Integrate the UI with the database and map engine

Week 4:

* Wiggle room (fix bugs, rigorously test application)
* Add flashy features (?)
* “Ship”

## 4.1: Actual Progress

Week 2:

* Research Flask, figure out how it works and how to put things on a local web page
* Research into Google Maps

Week 3:

* Build basic database elements
* Build two iterations of main application engine
* Discover way of using Google Maps and checking locational data
* Create flashy html templates, and css files

Week 4:

* Finish Database
* Finish location checking function and input validation
* Merge application engine components
* Combine separate program components
* Create Makefile
* Test application

## 4.2: Developer Logs

## 4.2.0: Conor’s Log

4/6/16: Try to figure out how flask works, look for ways to do a google map.

4/13/16: Found an extension to do google maps on flask, created small demo

4/15/16: Only way to validate if students addresses are within Safe Ride’s boundaries are to codify them. Found an extension that lets me get lat/long data from addresses.

4/17/16: Finished address -> True/False function. Required breaking Safe Ride boundaries into 26 regions with 104 long/lat vertices. Also made UO id validation function

4/20/16: Trying to integrate code with teammates.

4/21/16 – 4/22/16: Still integrating

4/22/16: Last minute touchups, compiling final documentation, building Makefile, and testing.

## 4.2.1: Zhibin’s Log

4/8/16: Initial reserach online for basic flask tutorial on how to make a web app

4/9/16: Set up a reservation page that shows textboxes that take student ID, name, and phone number and a "reserve" button that direct viewer to another page, with template from bootstrap.

4/13/16: Added textboxes for pick up time, pickup location, destination, and other comments.

4/15/16: Added function that pops up a new page to "reserve button"; started to work on .css file that modified page.

4/18/16-4/21/16: Integerate with work from teammates; aded a few pages and functional buttons that direct users to each of them; worked on web page modification and design.

4/22/16: Continue integration, finalize project

## 4.2.2: Jacob’s Log

4/5 - Following a tutorial online, I created a few simple webpages that allow Flask to accept text information and feed it into a database and return that information sequentially.

4/12 - Building off of the website, I created a few more pages to assume what the architecture of our final project would be. For example, I created a portal page that let me switch between what a student would enter and the results/map screen following that and what I'm ultimately going to spend most of my time on, which is the dispatch station. I've been tweaking variables and studying how this'll work.

4/13 - Created GitHub site and threw everything I have so far up there in a "preliminary\_backend" folder. I want to have a scrolling menu at the top of the dispatch page so that if you highlighted one of the entries, it would display everything in a table below instantaneously, but I have no idea what I'm doing and how to accomplish that.

4/15 - Added a few buttons to the dispatch page and added an ACTIVE variable to the database thinking I'll be able to switch this on/off so that I'll only see active requests on the dispatch page.

4/17 - Bailed on scrolling menu and went for a list of tables on the dispatch page. Since I'm running out of time, I figured it'd be better just to get it up on its feet first.

4/18 - I'm integrating what Zhibin did with the front-end with my page so it'll flow better. Also, I'm trying to go for requests that if you click on them, they'll open up with a few options to Approve, Remove, and so forth.

4/20 - Just a few changes here and there. Mostly for testing.

4/21 – Project integration, and tweaking.

1. The Riders: Jacob Brown, Zhibin Zhang, and Conor Tracey [↑](#footnote-ref-1)
2. *As opposed to the current system which is restricted to telephone requests.* [↑](#footnote-ref-2)
3. *As per Safe Ride’s request* [↑](#footnote-ref-3)
4. *The ID verification feature is limited in version 1.0 to checking the length, and first 3 digits of the inputted IDs. This is due to a current lack of access to the University’s ID database.* [↑](#footnote-ref-4)
5. *Necessary for flask to work properly* [↑](#footnote-ref-5)