Rideshare – EE464H Project Outline **January 3, 2008. Spring 2008**

Team Members:

Name	UTEID	Phone #	Email	Course
Yoni Ben-Meshulam	JZB62	(512) 633 0675	YoniBen@mail.utexas.edu	464H
Garrett Cooper	GC588	(361) 563 9723	GMCooper@mail.utexas.edu	464H
Patrick Lowry		(210) 861 3935	Lowry@ece.utexas.edu	464R

Supervising faculty member:

Overview of Rideshare

Rideshare aims to connect passengers and drivers seeking a "Rideshare" model of transportation, which generally optimizes the resources of travelers for mutual benefit. Rideshare will do this by minimizing travel time and travel distances, as well as taking into account the user preferences of both drivers and passengers. This Rideshare program positions itself as a service that takes the coordinating tasks and risks of seeking rides on bulletin boards (or similar models) by performing matching and route planning functions.

Description and Design Content

The objective of this semester-long project is to create the full rideshare web application, which can be used across the Internet to connect users interested in sharing rides. The application's service will be based on a matching algorithm that minimizes route lengths and times while maximizing the number of riders matched to drivers. However, the solution need only be workable: as our time is limited, and the problem complex, we will first concentrate on core functionality. The application will be interfaced through Google Maps, using a Ruby on Rails framework and a MySQL database implementation.

A good testing platform will be necessary in order to evaluate each module of the application. To accomplish this goal we must design the software with testing in mind, i.e. design-for-test. Each testing requirement must be quantified in terms of our final software architecture.

Our final product will require a server that can support our projected bandwidth and storage needs. The server must have an application server, preferably Apache, installed on it. Once we have the program running, and we are able to connect riders with drivers in an efficient and user-friendly manner, we will focus on ways to improve the application interface, aesthetic, and distribution.

Because this project will carry over from EE364D, we have done much of the high-level design work already. However, we will need to figure out exactly which functions we need to implement for each of our features, as well as some technology testing of third-party programs (such as the Google API), which we have only recently begun working with.

Deliverables

Our foremost concern is to provide the functional rideshare web application, which users can utilize to find and offer rideshares. We will offer documentation of the high-level software architecture, as well as the low-level module implementation. The actual code will document the programming structures and functions used for implementing low-level modules. At the end of the semester we will also provide a comprehensive demonstration of this application.

Testing
The implementation of each software module will include private functions specifically designed for testing. These will test the quality of each module based on the following criteria.

1. Functionality, i.e., the module does what it is supposed to. This will be most challenging for the matching algorithm, so we will place a significant focus on constructing edge cases. One case which we wish to support is picking up a passenger at points A and B, which lie close to the line the driver is taking from C to D, but A != C and B != D.
2. Performance, in terms of how quickly the function acts. Though the matching problem can be interpreted as N-P complete, we will narrow the problem's scope to execute in a workable amount of time. We have already found evidence of algorithms which run in polynomial time, as can be seen in our System Design document.
3. Stress. As we presume that our program will increase in functionality with the higher numbers of users, we need it to be able to support as many users as possible. We will simulate users in testability functions, and determine if the breaking point is high enough.
4. Security. Each of our software modules must be safe from exploitation and misuse. The best we can do here short of hiring a security expert is to try to hack and exploit it ourselves, ensuring that the private functions are not accessible by outsiders, and that each user's personal information is private.
5. Usability. Every function that relates to user interactivity must be intuitive and simple. The user must not be hindered by complexity or dead-ends, and we should be able to test this by watching people who are unfamiliar with the website use it.

Schedule
Within a month, we expect to have the basic website and database functionality complete. In the next month, we should complete the actual matching application, as well as specific data entry and internal/external user communications and any other website functionality we will need. Though we will be testing the application upon completion of every module, the remaining time will be used for extensive testing and perfection of the matching algorithm.

Individual Contribution
Though we each plan on contributing to all major sections of the project, each team member will have an area which they will 'own,' so to speak. Yoni will be in charge of the algorithm. Patrick will get the overall framework of the website working. Garrett will be in charge of the testing framework, as well as refinements made to the basic user interface and any visual design we might need.

Signed,

Yoni Ben-Meshulam
X_____

Garrett Cooper
X_____

Patrick Lowry
X_____

Professor Helpsus
X_____