Requirement document – team 3 Carpool Reservation Service

The following document outlines the requirements for our project of choice; the ‘Carpool Reservation System’. The following details are included: project purpose, intended audience, intended use, and project description. Additionally, there are also functional requirements, quality requirements, and process requirements.

Purpose: To offer an affordable, simple, and unique carpool experience for everybody in Barrie and, optionally, Orillia.

Intended use: To provide a carpooling service to those who wish to carpool with others.

Intended audience: People who wish to travel in a group to some destination at a cheaper price compared to public transit.

Functional requirements:

There will be a cloud-based relational database which will be used by our application to store all data which may be used in the future. This database will be hosted on a popular platform such as Microsoft Azure.

There will be several client-side HTML forms that will be submitted to our server-side for processing. When processing is completed, a success or error message will be generated and sent to the client depending on the status of the form after processing has been completed.

The application will contain, among others, a section which lists a table of all available carpool locations in Barrie and, optionally, Orillia.

When a user purchased a ticket for a carpool, the ticket will sent as a pdf to either their email or phone number, depending on the contact method of their choice. At the carpool departure the ticket will be checked by the system.

In the event a ticket is cancelled, the current date will be compared with the ticket purchase date to see if they have a difference of more than 24 hours. If the difference in time between the 2 dates is less than 24 hours, cancellation fees will apply to the user's payment method depending on the time of the cancellation. Anytime from 12 - 24 hours will charge 20% of the ticket cost, and 0 – 11 hours will apply 100% of the ticket cost.

Quality requirements:

The response time of the system must be less than or equal to 5 seconds

Bandwidth will be, by default, 1.0 Mbps per user.

Throughput will be measured by the number of successful transactions in our system. Therefore, our throughput must guarantee that at least 95% of the total number of transactions in our system are successful.

Resource usage will be handled server-side by whichever backend we choose to host on. Database connections, data queries, user profile information, Google Maps queries, etc. will he handled here and will be given as little as possible to a client to minimize the resources used.

The reliability of the system must be enough that it can handle at least 1000 transactions per second at any given time without decreasing in performance.

The availability of the system must be at least 99%. That is, the site should have at least 99% uptime. Specifically, below is the maximum downtimes for the framework:

Daily: 1 minute, 26 seconds

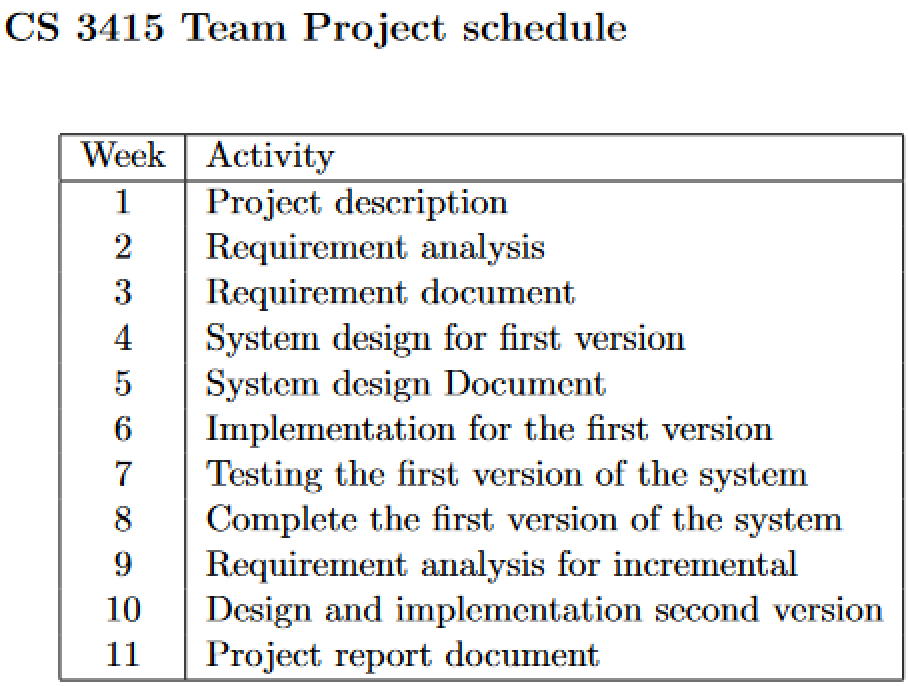
Weekly: 10 minutes, 4 seconds

Monthly: 43 minutes, 49 seconds

Yearly: 8 hours, 45 minutes, 56 seconds

Process requirements

The creation of our carpooling software system will chronologically follow the timelines stated in the ‘CS 3415 Team Project schedule’, as shown below:



CS 3415 Team Project schedule

Week	Activity
1	Project description
2	Requirement analysis
3	Requirement document
4	System design for first version
5	System design Document
6	Implementation for the first version
7	Testing the first version of the system
8	Complete the first version of the system
9	Requirement analysis for incremental
10	Design and implementation second version
11	Project report document

The list is subject to change if there is a change in functionality, or if the design process takes longer than anticipated. The project is equally split into separate important steps which include database design and connectivity, setting up our project according to the system design, implementing server-side data validation and security processes, and creating proper HTML forms. Team members will choose what functionality they will add to the project and keep a timeline revolving around this. The version control system chosen for this project is git, and by extension, GitHub. Commits will be made to the project repository by each team member, with a detailed descriptions of what was added. Our team will separate project development into multiple stages which will make us more focused on each part of the project equally. Saying this, we will focus more implementing the ‘Waterfall development method’ for our project.