Final Project Report

**iPark™**

Internet Programming I (ITEC 305 - M01)

Prof. David Cirella

**By**

Adam Israfil (1070289)

Ashean Silva (1131709)

Chengjun Dong (1078296)

Mohammed Rahman (ID)

**Summary**

This report contains a detailed discussion about the project iPark™, which is an online parking application. The purpose of this application is saving the valuable time of vehicles owners in New York city who live busy lives. The motivation of the project is to provide a solution for drivers who spend time on the streets looking for a parking space. These points will be widely elaborated in the introduction section of the report. Our first step towards developing this application was creating a github repository to allow for efficient code sharing among the group members. The client side of this project was designed using the HTML, CSS and Javascript languages. Furthermore, the app was locally hosted using a WAMP server with the table utilized PHPmyAdmin. The Discussion section goes more into detail on how we accomplished the final look and functions for our app. In the Recommendation section you will find the many errors and bugs we still have in our prototype. Due to time limitations we were unable to complete the full scope of features we wanted to implement including connecting to the database on page load to identify the taken parking spots based on what is in the database, more examples can be found in the conclusion.

**Introduction**.

New York City, the second busiest city in the world according to BBC rankings, and it is the only city in United States which gets an average of 220 vehicles per minute, or in other words, 316,800 vehicles per day, entering to the city from a different state (New Jersey), according to details published by Port Authority. Therefore, spending time on finding a parking spot, driving around the same block couple of times or waiting on the side of the road until another person takes his vehicle and leaves are the last thing a Busy New Yorker wants to do. It also causes to increase the traffic on the streets. Because of these reasons, a New Yorker who owns a vehicle is willing to pay even $40 per day to park their vehicles in the city.

Therefore by considering all these reasons, we came up with a business idea to provide a solution to this massive parking problem in New York City, and it is called iPark. The target market or the target customers of iPark are the vehicle owners and the drivers who looks for parking spaces in the busy areas of Manhattan City. Furthermore, the objective of iPark is to provide our customers online access to a website which will provide them informations about the parking spaces available in our parking building. In that way we promise to our customers that they can reserve a parking spot for that day or else to check out whether they have space to park in the building even before they leave the house or without actually entering or driving closer to the building for a time saving.

The user interface of this website is presented in a simple and understandable layout for any user. There will be three main functions (Buttons) in the home page which are park, leave, and report. Once the “park” button is clicked, it will direct the user to a new web page which contains the map of the parking slots in building with an indication whether it is reserved or not. Only if a spot is not reserved then website grant the user an option to reserve their spot by simply entering their first name, last name and the parking spot number that they wish to park their vehicle. Heading on to the next option, it is leaving which can be accessed simply by clicking the button “leave”. This option will allow the user to notify to the system that the user is taking their vehicle and the spot can be marked as available by simply entering the user first name and the last name. Moving on to the third option “report”, it is a function in our website which allows another user who already used the parking service to inform the system as he/she notifies an empty slot that is available but the user who left it did not complete the leave form in order to open that spot to another customer.

The client-side implementations and the user interface of this iPark website was built by the use of HTML, CSS, Javascript and JQuery. Then the server side was built using the Wamp Server with phpmyadmin, so that this will be a real time working website. Furthermore details about the implementing process, of iPark will be extracted and widely discussed by Adam Israfil, the lead developer of the spot selection page and the database.

Then, the further recommendations and the future implementations such as modifying the pages, adding new information and components, making the website more practical to use in order to increase the functionality and the security of the website will be discussed and explained by Chengjun Dong. On the other hand introducing the project presenting the idea of the project and its solution, explanation about the goals and objectives of the iPark will be done by Ashean Silva, the summary of the project was done by Chengjun Dong & Adam Israfil and then finally the conclusion of the project was done by Chengjun Dong and Ashean Silva. Mohammad Rahman created the slides for the in class presentation.

**Discussion**

The development phase of this project began with the creation of a github repo and an initial mockup of the main spot selection page. This initial mockup featured all the elements of what later became the leaving and report pages. The mockup elements were all hard coded initially in CSS. Each parking lot was a container that had a grid within of 2 rows and 10 columns. Each parking spot was manually coded in order to fill up the grid.

The next phase of development was deciding on the stylistic goals, in general how we wanted our forms to look on the site. We decided that we wanted something sleek and modern simple enough for any user to logon and be able to see what was available to them and what wasn’t. Once we had decided on the style we were approaching we realized that in order to achieve the vision of have the spot colors automatically change upon submission of the spot selection form we had to use jQuery. The initial plan was to alter the individual background-color attribute we had set for each spot (by default each spot is green meaning it is available). However, we realized this process would be a bit to difficult and changed our approach to have seperate CSS classes for taken and empty each storing respective background color attributes(red and green). When a user submitted the form the css class would change by means of testing to see the existing class, if the spot was taken the user would be prompted to select a new spot, if the spot was available the empty class would be removed and the taken class would replace it.

Now that we had the behavior on the submit ready that we wanted, short of actually sending the user information to the database, we needed to move away from the hard coded version of the grid. To do this we decided to utilize the jQuery on document ready function so that as soon as the DOM was loaded the grid would be filled up with parking spots, starting with the left column, moving right and then down to the next row. The original grid CSS class was then also replaced, now instead of the template rows and columns being defined in the CSS they would be tied to variables in the jQuery that would define the proper size. These variables also controlled the for loops that would fill the grid. This way the site admin can easily control the size of the lot and quickly adapt to any size lot.

The last phase of development was setting up the database. After researching, we settled on a locally hosted WAMP server that ran phpMyAdmin. The information we wanted to store were the user's first name, last name and the spot they selected and each needed to have its own unique server ID so that it could easily be referenced later on. The server would be locally hosted on Adam’s laptop such that it can also be used for the purposes of the in-class demonstration. With the database setup the jQuery now had somewhere to send the information. To communicate with the database we wrote a php script that would establish a connection to the server and upon successful connection would add the first name, last name and selected spot to the members table. To connect the php and form submission we tied it to the same onsubmit function that changed the color of the selected spot. To create the request we used an ajax asynchronous call, meaning that the call executes without hearing back from the backend/database that it worked, but we don't know when we'll hear back. The call is directly told to use the php script and that it is meant to post the information that the user provides in the signup form. There are three request handlers: done(tells the user that their spot was successfully reserved), fail (alerts the user that there was some sort of error and this error is also posted in the console log for debugging purposes) and always (prints in the console that the request was made).













**Recommendations**

Even though the project runs without any error, there are still space for improvement. For the future, the website page will looks better, and apply the reporting page css to other pages as well. The “contect us” and “policy” button on the bottom of the homepage will also redirect to other pages that we will create. It will add a timing where user can only park for certain numbers of hour, after time passes, there will be additional charge, and will be towed away if the time is executed too long.

For the parking page, after the user get a spot, the data is stored in the database, and user should get a random code that is generated by the machine, and whenever user wants to leave the spot, he/she needs to provide their informations as well as the code that matches the database, otherwise the spot will still registered. Also, the statues of the park space should remain the same even after the user refresh the page, which means user can see which spot is taken, the color of the spot should not go back to all green after refresh the page.

For the reporting page, whoever wants to report an issue, they have to be registered at parking space in order to protect security where not anyone can access and change datas. For the future, the application will check if the input data are matches database, as well as for leaving page.Whenever after a report has been confirmed, the color of the parking page should be change.

**Conclusion**

In the 21st century, due to the decrease of poverty, more and more cars are affordable to most of the families. However, because of the number of cars increases dramatically, especially in the city, parking lot is getting harder to find. Therefore, we came up an idea where users can visually see if there are any parking spots left, and register a spot if they would like to. Our project is an application, called iPark™, allows user to find and reserve parking spots in parking structures. In conclusion, we are trying to make an application that can help our New Yorkers get easy access to find a parking lot, reduce all unnecessary times to find the treasure.

Overall we wanted to create an app that’s both easy to use, navigate, create, and implement. iPark™ was designed to make it easy for people in busy cities find parking in parking structures faster and cheaper. Out app not only allows people to find open places to park but also allows users to reserve a spot for parking. The current prototype is just a basic working prototype to demonstrate the core functionality of reserving a parking spot. We can also implement a rating system so you can see how good a parking lot is. Later on it can be modified to be secure and handle payment as well improve the login system to make accounts secure. Stylistically, our goals is to have all the forms on the site be the same as the current report page(image can be found in Discussion). There are so many directions we can take this project and it can definitely make people's lives a lot easier.