# Discussion

Throughout the development process of the KTCS service the team experienced a variety of problems, but the team was successfully able to overcome them. The most prominent issue experienced was the steep learning curve of PHP and HTML at the beginning of the project. None of the members on the team had worked with PHP before and this resulted in quite a bit of time at the start to be devoted to learning the basics of PHP. This included learning how to pass values between forms, embed HTML into PHP, retrieve SQL results and display results to user. The team was able to solve this issue by starting the project early. The team started to work on the coding aspect of the project almost immediately after the Phase 1 report was due. This gave the team ample time to learn the required tools.

In regards to PHP one of the biggest issue was figuring out how to properly use the SESSION variables. The team used SESSION variables to pass around prominent information and ensure it persisted between forms. Some of the values passed through SESSION variables included the currently logged in memberID as well as their name. The purpose of using global SESSION variables was to optimize the product to remove the need for redundant queries (otherwise would need to query KTCS\_Member to get user info on every form). The team ran into issues with the SESSION variables later in the development cycle as it was noticed that at times the SESSION variables were not updating with the correct values, even though they were being set. This issue was fixed by ensuring that the cache and cookies were cleared before starting the KTCS service.

Debugging in PHP was a common issue experienced by all members of the team. To simplify debugging and to ensure consistency between members when debugging the various php forms, all the SQL statements executed were echoed along with every row’s value for every column of a given table. Debugging was one of the most time-consuming aspects of the project as it would seem that the query executed without errors, but then when looking at the database the object would not be inserted/updated. Therefore, echoing all the SQL statement results ensured the team could see what the issue was. One aspect where debugging was not possible was when populating the database using the KTCS\_DB script. When inserting values to the tables through the script at times it was noticed that the database was not populated correctly. The KTCS\_DB did not give any debug info so the team had to solve these issues by explicitly copying and pasting the SQL query into PHPMyAdmin to check the error that it was giving.

Finally, with the high volume of php files produced for this project it became hard to track them at times. At times the team had a standalone form (ie. monthlyInovice.php) that was merged as part of the homepage, but the monthlyInvoice.php form was not deleted from git. This resulted in confusion between members when it came to editing. The team overcame this issue by consistently refactoring the files in git and removing the files that are no longer in use.

As for technologies used in this project the team used HTML, PHP, CSS and Bootstrap for the code as well as GIT for file tracking. The reason why the HTML, PHP, CSS and Bootstrap suite were selected is due to their ease of use in comparison to other tools such as JavaScript or building a Python based server. The team members had very limited knowledge of HTML and essentially no knowledge of PHP or CSS. The inclusion of Bootstrap for the HTML made styling and constraints easier. GIT was used for file tracking and for revision monitoring. GIT allowed all members to maintain the same view of the code at any point and if an issue with a given members code did occur it was very easy to isolate it and revert to a commit where the feature was working. PHPMyAdmin as well as XAMPP were used to provide the MySQL database to the application.

One of the most important design decisions the team made was spending a lot of time designing the ER schema. The ER schema developed ensured that it would be able to meet all the functional requirements using 9 tables with a combination of weak/strong entity sets, primary and foreign key relationships, various cardinalities (such as many to one) and integrity constraints. By stressing the importance of the ER schema, it allowed the team to easily convert the ER schema into relational algebra which was then converted into SQL to create a strong foundational architecture. This allowed the team to more seamlessly perform the various queries involved in this project as the underlying DB design was designed to ensure these queries were handled correctly.

Another design/implementation detail the team focused on is modularity. Firstly, the team solely created one MySQL connection object that was passed to every form through the use of the include PHP command. This allowed the project to only contain one database connection object that could be accessed from any form instead of having to create a separate connection object in every form. In addition, for every user interaction form there was an associated handle form that would be called when the user hits submit. The purpose of having separate handle forms and not having both user interaction and handle in one form is for modularity. It allowed the team to separate user interaction from functional SQL code. Moreover, with this approach the user interaction form code could be altered without altering the functional SQL code (in the handle form).

If the team had the chance to go back the team would change a few things. Firstly, the team would research more into using SESSION variables as a global way of passing variables between forms. The SESSION variables were used throughout the project, but it did lead to issues and at times a SESSION variable would not update to a new value. By researching more into this area, the team would understand it better and ensure that such a problem did not happen. In addition, the team would protect against SQL injection by using prepared statements. Currently, the project developed does not use prepared statements and thus it is susceptible to SQL injection. Finally, the team would try to host the website on a server rather than running it locally to create a more coherent development environment. If the website was hosted on a server, the team would be able to develop on this server and thus all team members would always be viewing the same DB and code (instead of local copies).

Overall, upon completing the project, the team learned a lot about HTML, PHP, CSS, Bootstrap, PHPMyAdmin, MySQl and GIT. The team came to understand the complexity and potential of these tools and gained some experience using these tools. The team made various important design decisions that guided the team throughout the development process such as stressing the importance of the ER schema and a modular form design. The team also ran into several issues such as tracking old files that are no longer in use and the learning curve for PHP, but the team was able to overcome these challenges. In conclusion, the team is proud of the work it accomplished with the KTCS service and hopes that one day this service will become a viable product that all students and residents of Kingston can use to gain more flexibility and convenience.