# Green Apple Pay 2020 - T661 Critical Notes

IMPORTANT: Use existing materials and documents generated during the course of your development work to extract the pertinent information to complete these questions.

## 242 – What technological obstacles did you have to overcome? 350 words max – 50 lines of 78 characters

The state of technology at the beginning of the project had several shortcomings and/or limitations, as well as technological problems and unknown elements, as described below.

Typically we are faced with using a tried and true e-commerce model for purchasing products, but we quickly realized that this was not scalable, and inadequate for the number of unique edge cases our platform required. We hypothesized we could potentially develop an event and schedule-based hybrid application model to accomplish this. We commenced work to overcome obstacles related to the workflow customization process within the typical checkout cart process. We analysed the process and attempted to develop an approach, such as event-based modules, which would allow adjustment of the order status and data within the cart, and allow these events to be created, scheduled, and repeated.

It was uncertain how we would integrate with many reporting, device, and system meta data processes and tools. We were uncertain how to develop a middleware layer to achieve this, while maintaining control and security of data, as well as overcoming transaction performance issues.

## 244 – What work did you perform in the tax year to overcome those technological obstacles? 700 words max - 100 lines of 78 characters

In the 2020 fiscal period, we continued to overcome our e-commerce model limitations by setting the framework for testing integrations for a better payment model. Development of a payment transaction abstraction needed to be built, as there were too many payment transaction types among too many nodes to be secure using known technologies. Our hypothesis is that we can allow this payment process to be simpler with less steps; we can potentially integrate a wallet technology within our system as a payment clearance layer. We think it could be highly secure and only available to trusted parties. In this fiscal period, we would develop the framework so that we could complete tests and develop an experimental integration path extending from payment APIs, in an attempt to overcome these limitations and advance our toolset.

Through October and November 2019, Cappers Applications Inc, our sub-contractor, continued to develop an approach to differentiate the typical e-commerce shopping cart and payment integration processes, which required product item data deletion, as well as the integration of a tokenized payment process into the e-commerce model. We developed and extended a code base to process credit payments only. In November, we hypothesized we could overcome existing payment API limitations by extending various debit and EFT library capabilities. In December 2019, project resources were redirected, so the project was shelved, and no further SR&ED development was done until July 2020.

In the last month of the fiscal period, the project was started again in July 2020. Our new contracted architect, Timur Imnaishvili, set out to continue to attempt to build a test bed for our limitations, by introducing an approach built on the Phalcon framework.

Phalcon © is a full stack PHP infrastructure that operates as a web server extension. The application itself is written in Zephir and C, with the latest release (December 2019) written entirely in Zephir. The application is installed compiled as a binary, so no C or Zephir coding is required. It is our hypothesis that it will allow us to tokenize all data, including payments other than credit. It operates on the server side, thus maintaining the backend processing that is the philosophy of the rebuild.

It is our belief that the Phalcon infrastructure will operate as a single extension and provide us with the secure payments we need for all forms of payment. It takes care of automatic loading and is extremely fast. This advances our constant goal of superior performance and backend only processing. To arrive at this decision we evaluated the merits of other backend integration, like Symfony ©, Laravel © and a couple others, but they are, in our opinion, over engineered and have a likelihood to create new, as of yet unanticipated problems. Phalcon is extremely simple and straightforward.

By using Phalcon it is anticipated that client participants, schools in this business model, will be able to apply the Phalcon PHP coding required for integration. It must be remembered that we are not just building something to work for us; we are building a platform clients have to integrate with by coding PHP.

There is no certainty that this new model with Phalcon will work completely. Build and testing is scheduled for the next fiscal year, but we do feel secure that the decision we have made with Phalcon will provide the best chance of meeting all of our edge cases and the design philosophy of backend processing, keeping everything away from the browser, while maintaining scalability and speed.

## 246 – What technological advancements were you trying to achieve? 350 words max - 50 lines of 78 characters

The technological objective of the project is development of a Data Brokerage Layer (DBL) for school e-commerce systems. We realized this layer became necessary in dealing with the voluminous amounts of information on abstract school networks. This layer works as an intermediary that provides data and service integration between school systems, a payment gateway, and any school board systems. We realized that student and payment data could be processed much more efficiently if there is a layer that can work as data broker and can process events and purchases ad-hoc by various user types. We were also able to develop support for better data transaction services, as we were able to accomplish more sophisticated, time-delayed event updates and scheduling related to future-compensating transactions.

In this project, we were trying to achieve development of an architectural framework which focuses on the exchange of data to facilitate the financing and payment aspects of school’s cash transactions, related to product and scheduled event purchases. The work done advanced the existing state of knowledge in the field of Information Technology, specifically in the E-Commerce Pattern methodologies.

This development represents a technological advancement in the e-commerce workflows, as existing technology and knowledge base level at the onset of this project was limited, and we realized that the typical e-commerce order workflow would not function for our model. Usually e-commerce store approaches are comprised of a combination of design patterns, such as model-view-controller (MVC) design pattern, factory pattern (for maintaining orders and bill generation), and the observer pattern for updating inventories and product data. However, alone these would not be satisfactory in the context of school business, purchase events, and application logic: purchase processes would be more complex. Project or campaign management API’s existed, but they were not suitable or reliable to integrate to meet our edge cases. There is a low certainty of integrating a scheduled, repeatable, events-driven structure into any e-commerce model, especially when considering scalability. Further, there is a lack of system security, reliability, standards and communication protocols within the school environment. It was difficult to integrate the Internet and E-Commerce software with some existing applications and databases. Uncertainties required experimentation to determine an advanced process.

The work done resulted in development of an architectural framework which focuses on the exchange of data to facilitate the financing and payment aspects of a school’s cash transactions. Our approach allows schools and companies to link their internal and external APIs more efficiently and effectively, and manipulate data safely, efficiently, and more secure.