SmartShop: A Merchandise Order and Deliver Application using Mobile Devices

Xavier Minaya Ruiz
Master in Computer Science
Advisor: Othoniel Rodriguez, PhD
Electrical and Computer Engineering & Computer Science Department
Polytechnic University of Puerto Rico

Abstract *— In our modern society, time management is very crucial. We spend most of our time on important events in which sometimes we put aside necessities such a going to the supermarket for our nutrition or pharmacy for our health complications. The idea of SmartShop is to provide users using a mobile device to obtain certain products that they need without having to spend much time of their day to go to the store. SmartShop allows the opportunity to provide delivery to their home. Car drivers will access SmartShop application to obtain the delivery information of the orders that have been assigned to deliver them to the clients.*

Key Terms *— Delivery, Mobile Device, Order, Product Catalog, SmartShop.*

INTRODUCTION

Technology has been exponentially evolving every year, in which has helped society on daily basis routines or for new discoveries. The usage of technology in our society has helped improve quality of life in which has provided opportunities to fulfill our necessities with ease. We can find at least one application or service that resolve any difficulty that we can encounter.

Unfortunately, when we must purchase necessary products we must plan ahead on our agenda to do so, spending a lot of time that we could benefit on something else. Sometimes we must postpone in purchasing our basic needs because we don't have enough time to go to the store. This is where SmartShop comes to fulfill this modern gap using innovative technology. SmartShop prevents users in spending too much time in purchasing products and have them delivered a specific location.

In SmartShop, user will have access to a product catalog that will satisfy their needs to purchase products at the reach of their mobile devices. Users will have the option to have their purchased products delivered to them or pick them up on purchasing site. If the users decide to have their order delivered, car drivers that has been approved by SmartShop will be able to have them assigned for delivery. Car drivers will access SmartShop Delivery application to obtain the delivery information of the orders that have been assigned to deliver them. Administrative users will manage product catalog and delivery assignment to maintain the system consistent for operations.

While SmartShop offers a convenient way for users to purchase products from a mobile device, it will give opportunity for any company to use the platform to reach out to the consumers by offering their products through SmartShop. SmartShop system is design as a product so companies could have a merchandise and delivery system to enhance their business. This would allow company to explore even more the target consumers that the want to reach, allowing to become more competitive within their market and have an opportunity to expand their products.

REQUIREMENTS

As displayed on Figure 1, the SmartShop system consists of three applications: SmartShop Client, SmartShop Delivery and SmartShop Administrative Portal. In SmartShop Client, users are be able to browse the product catalog to create orders that will be delivered to a particular location by a car driver or pick up by the client. Users could greatly benefit in purchasing products from any company that are taking advantage of the

SmartShop system to promote their products and offer delivery service to their customers.

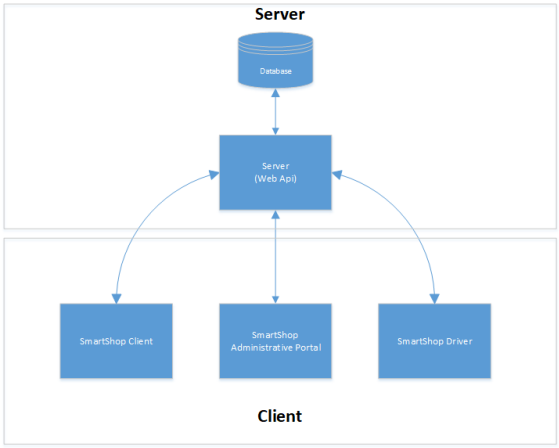

Figure 1
SmartShop's system

While browsing to a product, user may choose a product with different attributes while it is still available for purchasing. Also, products that are candidates to be purchase will be included into a shopping cart allowing users to save their products to purchase them later. The following requirements are implemented in SmartShop Client application:

- Browse through Product Catalog
- Maintain items in shopping cart if order hasn't been placed
- Place orders on items on the shopping cart
- View order's information
- Manage payment methods and delivery addresses
- Obtain user's current location for delivery
- Push notification for delivery's status

In SmartShop Delivery, the users are the car drivers assigned to deliver the order to the users of SmartShop Client. Car drivers will see the delivery and order information to make the arrangements to deliver the orders. When the car driver is completing a delivery, they must enter a delivery confirmation to provide delivery integrity for the clients. To become a car driver for SmartShop, candidates must submit a series of required documentation for eligibility. Only users that has been certified by SmartShop administration are allowed to carry on with the deliveries. The following requirements are included in SmartShop Driver application:

- View delivery's information
- View client's address information on map for order delivery
- Manage software driver's account
- Sign up for delivery services
- Delivery confirmation
- Software driver's account validity for delivery services
- Push notification for delivery assignments

In SmartShop Administrative Portal, administrative user manages the product catalog, view delivery information and users within the portal to carry on the operation of SmartShop. Administrative users are responsible to provide enough products that will allow users on SmartShop Client to purchase. Managing delivery information allows administrative user to assigned orders to car drivers and change delivery option if necessary. Candidates that wants to become a car driver are certified by submitting a series of required documents, in which administrative user verify for validity. The following requirements are included in SmartShop Administrative Portal application:

- Manage products' information
- Manage software drivers' information
- Manage deliveries' information
- Manage portal's users information

DESIGN

The applications that will be used by the users are designed using client-server architecture, in which all the information will be provided by the server as shown on Figure 2. SmartShop Client will display all the products available configured by administrative users on SmartShop Administrative Portal which will allows them to create orders. SmartShop Driver will be able to see the orders that they should deliver assigned by to them, all coming from a single data repository managed by the SmartShop Administrative Portal.

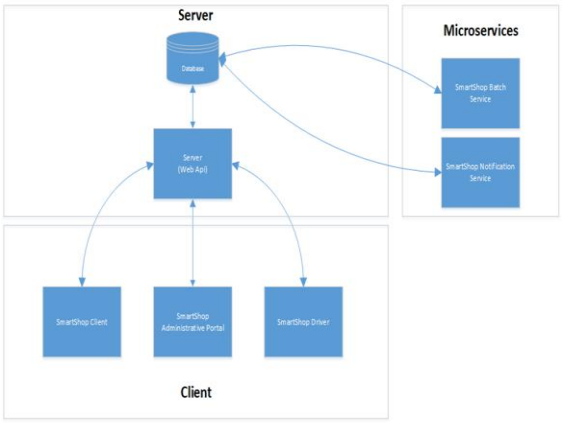

Figure 2
SmartShop's Architecture Design with Internal Components

Internal system components are designed as micro services, in which they execute independently accessing data from the database to execute their corresponding processes. Using micro service allows these processes to be installed on any environment, removing dependencies from any other process. The internal system processes will handle sending notifications and any other batch processing to ensure that business rules such as order delayed and software driver's validity are detected on SmartShop system.

Since SmartShop is developed as a product, each company would have a standalone version in which users would install the company's application version of SmartShop. Each company would have an administrative users that are responsible in configuring products, approving candidates that would like to become a drivers for delivery services, and delivery assignment for orders that would be manage through the SmartShop Administrative Portal.

DEVELOPMENT METHODOLOGY

The development methodology for SmartShop is Agile. Agile is a software development method that proposes an adaptive mechanism in embracing any changing requirements as the project is in progress. It is a reaction to the failing of more rigidly practice that the waterfall software development cycle has been proposed throughout the years. The waterfall process doesn't comply very well with change requests.

With Agile, it uses an approach in which welcomes any changes and allows to adapt to any changes that will occur. It allows changes at a process level, and developer must be able to implement a source code that is adaptable for such changes. It is highly recommended to uses SOLID principle [1] when working in an agile project.

SmartShop system adopts Agile as the software development methodology and in software implementation uses the SOLID principle. Using the Agile methodology, features on each module are implemented in iteration that allow to have a dedication in completing assignments within a timeframe. On Figure 3 display the features implemented per iteration for SmartShop Client. Also, future features can be integrated with ease since the module are implemented using the SOLID principle, reducing major breaking changes in development.

While developing against an interface abstraction that enforces the SOLID principle, development on individual components can continue without depending on other modules since they can be mocked. This prevents development to be delay due to dependencies.

Iteration 1			
Feature	Description	Status	Developers
Sign up	User will create account to login into SmartShop Client	Completed	Xavier Minaya Ruiz
Forgot Password Wizard	User that has an account will be able to recover by going through the forgot password wizard	Completed	Xavier Minaya Ruiz
Iteration 2			
Feature	Description	Status	Developers
Browse product category	User will be able to see product category of product that are available to purchase	Completed	Xavier Minaya Ruiz
Browse product	User will be able to see available product to purchase	Completed	Xavier Minaya Ruiz
View product	User can view a product information such as attribute and quantity to purchase	Completed	Xavier Minaya Ruiz
Iteration 3			
Feature	Description	Status	Developers
View deals on Home Page	When user logins, on the home page they will able to see available deals (product) to purchase	Completed	Xavier Minaya Ruiz
Add product into shopping cart	User can add item into shopping cart when visiting a product	Completed	Xavier Minaya Ruiz
View pending items on shopping cart	When user logout and visit the app, the must see their pending items on shopping cart to place orders	Completed	Xavier Minaya Ruiz

Figure 3
Agile's Iteration for SmartShop Client Development

IMPLEMENTATION

There are three application that composes SmartShop as a system. Each one of them represent a major role for SmartShop to be functionally operational.

SmartShop Client

SmartShop Client is a mobile application that can be download by any device that supports Android and iOS operating system. The technology used to implement SmartShop Client is Xamarin [2]. Xamarin is a cross-platform mobile framework backed up by Microsoft that allows to write on any .Net programming language to implement native mobile application.

On SmartShop Client, the design pattern MVVM was used to allow adoption of SOLID principle and maximum code reuse for both iOS and Android platform. The MVVM framework used is MVVMCross [3], a framework that handle the majority of platform specific logic and allows developer to concentrate more on business logic implementation.

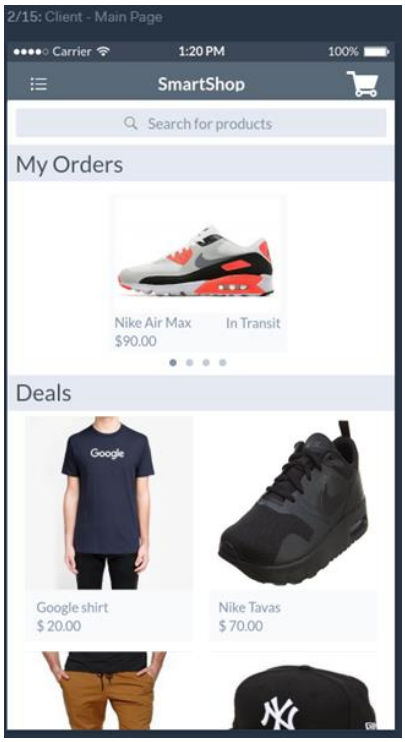

Figure 4
SmartShop Client

SmartShop Delivery

SmartShop Delivery is a mobile application that can be download by any device that supports Android and iOS operating system. The technology used to implement SmartShop Delivery is Xamarin.Form. The difference of Xamarin and Xamarin.Forms is that Xamarin uses the platform API (Java SDK for Android and Objective-C/Swift for iOS) translated from .Net to develop application for individual platform and Xamarin.Forms uses a SDK that automatically target each platform without need to go into detail or including a lot of logic for each one of them. Compare to SmartShop Client, this application have a simple UI design that proposes to offer the necessary functionality for car drivers carry on their responsibility.

On SmartShop Delivery, the design pattern MVVM was also used to allow adoption of SOLID principle and maximum code reuse for both iOS and Android platform. The MVVM framework used is Microsoft Prism [4], a framework that has an excellent MVVM design for Xamarin.Forms, adaptability of module using dependency injection, messaging system for component to communicate with each other, and a great loosely couple navigation service within page.

Microsoft Prism provides additional functionality to enhance memory management in applications to remove the necessity in manually calling the garbage collector. The following functionality not offered by Xamarin.Forms that Microsoft Prims provides are:

- Allows developer to write logic when navigating into pages
- Safely disposing messaging subscription to prevent memory leaks
- Display views depending on platform without writing specific logic
- Automatically mapping of view model for each view based on naming convention.

For the IoC container on SmartShop Delivery, the Ninject framework was selected due to with its flexibility and services instantiation managements of the objects.

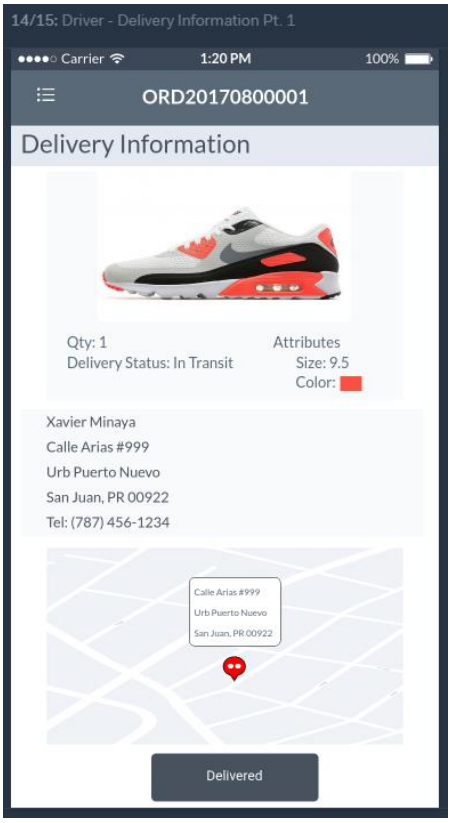

Figure 5
SmartShop Delivery

SmartShop Administrative Portal

SmartShop Administrative Portal is a web application in which administrative users will have access to manage products, deliveries, software drivers and users' information within the SmartShop system. The technology used to implement SmartShop Administrative Portal is ASP.NET MVC Core [5]. ASP.NET Core is Microsoft's lasted web technology that allows to deploy web application in multiple operating system aside of Windows, allowing cross-platform capability. ASP.NET Core offers flexibility on including functionalities into the HTTP request pipeline to add feature without affecting the application's structure. Functionality such as security, hosting static files and ETC are configured with a couple lines of code in which the application adjusts accordingly.

ASP.NET Core support dependency injection in which initially proposes developers to adapt SOLID principle into the application. All services are injected into the logic, allowing loosely coupled and unit testable code within the project.

TESTING

SmartShop modules have been developed using SOLID principle. SOLID allow to implement code that is adaptable and unit testable. In order for code to be unit testable, it should depend on an abstraction and its services should be injected through construction. Two of the design patterns that has allowed SmartShop to become unit testable is the interface segregation and dependency injection.

Every service in SmartShop implements an interface, in which concrete implementation is abstracted away from its consumer. This remove dependency in concrete implementation and prevents breaking changes in the code, allowing the code to be adaptable and testable.

Unit tests are done using a testing framework called NUnit [6]. NUnit is an open-source framework that has been widely adapted within the community because if its ease integration with multiple IDE and simple API to write unit test on any logic. The approach of SmartShop is to write unit test on every logic to be integrated with the application per say. This will greatly reduce bugs on individual components within every module.

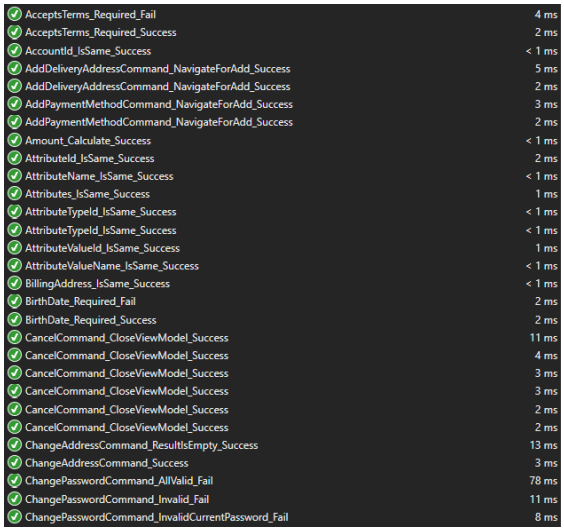

Figure 6
Unit Test using NUnit

SmartShop has initially adopted Test Driven Development (TTD) [7], thanks to using the SOLID principle. Doing so proposes a huge advantage in maintaining a testable and adaptable code that can include any feature, while ensuring integrity on existing code. Every logic implemented on SmartShop has its associated unit testing, in which the starting points for developers is the unit test. By using the Red-Green-Refactor technique in Test Driven Development, code is included as a unit test into the module within SmartShop.

CONCLUSION

SmartShop offers an innovative way for users to purchase their necessary products from an accessible mobile device without the need to take part of their time to do so. It offers adaptability on any person that has a busy lifestyle and prefers to have their daily chores handle with ease. Using modern technology, SmartShop fulfills a gap in time management that we encounter when purchasing our daily basic products.

Even though SmartShop's main intention is to provide an easy way for users to purchases their necessary products, it also provide company that would like to enhance their merchandise and delivery service by having their customize version of the system.

SmartShop would be a great associate for company that excels on taxi/car services such as Uber. Associates of Uber can receive a commission from SmartShop administration by offering delivery service for orders of SmartShop clients. This opportunity will allow SmartShop to expand its business internationally, giving an opportunity on foreign countries to take advantage of purchasing products from their mobile device with spending time in doing so.

As technology progresses and necessity demands, SmartShop could excel it delivery capabilities by using FAA's regulated drones to allow doing aerial delivery and enhancing delivery service beyond the user's expectation. This type of delivery would allow SmartShop to reach place that is not very common for a software driver using a vehicle or bike and provide an extensive reach to delivery products to customer.

REFERENCES

[1] S. Oloruntoba. (2015, March 15). *S.O.L.I.D: The First 5 Principles of Object Oriented Design* [Online]. Available: https://scotch.io/bar-talk/s-o-l-i-d-the-first-five-principles-of-object-oriented-design.

[2] Xamarin. (2017, January 14). *Mobile App Development & App Creation Software* [Online]. Available: https://www.xamarin.com/.

[3] MVVM Cross. (2017, December 11). *MVVM Cross 5.6* [Online]. Available: https://www.mvvmcross.com/.

[4] Microsoft. (2017, September 21). *Prism for Xamarin.Forms – Overview* [Online]. Available: https://prismlibrary.readthedocs.io/en/latest/Xamarin-Forms/1-Getting-Started/#overview.

[5] Microsoft. (2016, June 7). *Introduction to ASP.NET Core* [Online]. Available: https://docs.microsoft.com/en-us/aspnet/core/.

[6] R. Prouse. (2017, April 18). *What is NUnit?* [Online]. Available: http://www.nunit.org/.

[7] S. Wambler. (2018). *Introduction to Test Driven Development (TDD)* [Online]. Available: https://agiledata.org/essays/tdd.html.