

**Analyze user daily spending time to inspire their self-discipline
behavior by designing and developing applications**

Creative Capstone
for
Master of Science
Information and Communication Technology

Wei Lun Huang
University of Denver University College
July 01, 2017
Faculty: Timothy E. Leddy, MBA
Director: Thomas Tierney, PhD
Dean: Michael J. McGuire, MLS

Abstract

This creative capstone illustrates the design and development of an application prototype that analyzes the time spent by the users to inspire self-discipline. The application is called DiSpend, symbolizing *Discipline Your Spending*. It has a responsive design and includes both a mobile and a desktop version. The functions focus on the primary source of time spent and uses it to visualize different types of charts for the user. The interface allows the user to set goals about how much time they expect to spend on each task. DiSpend is a time management application. DiSpend aims to assist users to predict their time spent on different tasks, so people can adjust their daily activities to have better-organized plans.

Table of Contents

Abstract	ii
Table of Contents	iii
List of Figures	v
List of Tables	v
Objective	1
Introduction	1
Creative Capstone: DiSpend Application	2
Full-stack design and development	2
Usability	3
Application Flowchart	3
Flowchart Keys	4
Flowchart 1 - User registration and login	5
Flowchart 2 - Application Processing.....	6
Flowchart 3 - Data Visualization	7
Unified Modeling Language (UML)	7
UML 1 - Use Case Diagram.....	8
UML 2 - Sequence Diagram	9
Prototyping Model	11
Function 1 - Splash Screen	11
Function 2 - Number Keyboard.....	12
Function 3 - Progress Bar	13
Function 4 - Chart Report	14
Analysis of the Proper Use of Native App, Hybrid App, and HTML5 App Methodologies	16
Native App Development.....	17
iOS platform	17
Android platform	18
Windows platform	18
Hybrid App Development	19
Apache Cordova	19
HTML5 App Development.....	19
Responsive Web Design	20

Progressive Web App.....	20
How to choose a right approach?	21
Performance, Speed, and Cost	21
Native Performance	21
Native Cost and Development Speed	22
HTML5 and Hybrid Advantages	23
The Future of Mobile Development	25
Security Overviews.....	25
Code Protection	25
Network Security	26
Encrypt Authorization	27
Security Architectures.....	28
The Final Decision among Hybrid or Native or HTML5.....	28
Reflection: Mobile Application Development	30
Why I chose this project?.....	30
Foreword.....	30
What I have learned from University of Denver and used for this project	31
Mobile Design	31
Coding Skills	31
Mobile Application Development.....	32
Design Modeling	32
Information Security	32
Business and Information Technology initiatives	33
Skills and Knowledge Demonstrate	33
Agile Modeling	34
CSS3.....	34
JavaScript	34
Github	34
Web Hosting.....	35
Cordova Android	35
Final Perspective	36
References	38

List of Figures

Figure 1. Flowchart key illustration.	4
Figure 2. Application flowchart for user login and sign up.....	5
Figure 3. Application process flow for the main page.	6
Figure 4. Data visualization flowchart for viewing the report.....	7
Figure 5. Use Case diagram.....	9
Figure 6. Sequence diagram for describing the relationship between the user, application, and the database. Also, for the process of registration, login, main page and logout.	10
Figure 7. Splash screen.	12
Figure 8. Number keyboard.	13
Figure 9. Progress bar.	14
Figure 10. Chart report.	15
Figure 11. ComScore report, Number of global users.	16
Figure 12. Mobile app development estimated timelines.	23
Figure 13. DiSpend hybrid application that built with Cordova.	36

List of Tables

Table 1. The Mobile App Comparison Chart: Hybrid vs. Native vs. Mobile Web.	22
Table 2. Developer Salaries in the United States.....	23

Objective

DiSpend is a time management application. Many people frequently spend time in an inefficient and unorganized manner. As a result, this creative capstone project focuses on designing and developing an application that can store how people spend their time. After collecting data for a time, the application will analyze and display monthly statistics. It will also provide positive feedback to users to inspire self-discipline in their life. The statistics should report how users spend their time and let them develop healthy habits. Therefore, DiSpend can motivate the user to achieve his or her goals. The objective is to inspire users' self-discipline, record how they spend their time, and give them the advice to increase productivity.

Introduction

DiSpend is a time-tracking application that is built with HTML5 programming. DiSpend deploys on an online shared hosting and can use Apache Cordova platform to transfer to a hybrid mobile application. Its purpose is to help people spend time wisely.

The following section "Creative Capstone" describes what full-stack development is, and the full-stack technology skills used in the developing process that include: usability, application flowchart, unified modeling language and prototyping model. The "Analysis" section aims to clarify the differences approaches to build a mobile application between native app, hybrid app, and HTML5 app. The "Reflection" section shows why I chose to study mobile application development and what skills I used for creating this project.

Creative Capstone: DiSpend Application

Full-stack design and development

The layers of a full-stack application include both the physical and the logical layers that allow the application to operate. The physical layer includes the servers, the network, and the hosting environment. The logical layer includes data modeling, APIs, the user interface, and the user experience (Gellert 2012). When a designer or developer wants to implement an application, the process should involve six steps: design, test, develop, deploy, document and publish. Also, the process should take into consideration the development expertise required for each layer of the development stack.

DiSpend is a full-stack application. The design considers three aspects: usability, application flowchart, and unified modeling language. The usability aspect of the design takes into account usability principles to create a good user interface. The usability aspect of the design also describes how to integrate the Bootstrap framework to achieve Responsive Web Design (RWD). The application flowchart aspect of the design presents three processes including user registration and login, application processing, and data visualization. Each of these processes has one flow chart to show how the application works. The unified modeling language aspect of the design displays the use case diagram and the sequence diagram. Both diagrams describe the relationship between the user, application, and the database.

The development focuses on the creation of a prototyping model. The prototyping model describes four primary functions of the application that are used in DiSpend. The four types of functions, i.e. splash screen, number keyboard, progress bar, and chart report, have been completed as a working prototype.

[A link to the working prototype can be found here.](#)

Usability

Usability is important to consider when first designing an application. A proper usability design can provide the user a good experience, which means more users are willing to use the application. Thinking of the user interface(UI) principles, a good UI has high conversion rates and is easy to use (Linowski 2014). It is because nobody wants a frustrating experience, such as viewing a small font on his or her mobile or tablet devices. Also, typing a long list of form fields for registration might cause a bad user experience. Moreover, applying responsive web design (RWD) approaches can optimize the content for better usability. RWD makes the desktop's content automatically fit the screens on mobile, tablet, and laptop versions.

Responsive Web Design (RWD) has been commonly used in web. Because DiSpend uses a web platform, it can apply Bootstrap to realize the RWD function. Bootstrap is a popular HTML, CSS, and JS framework for developing a responsive, mobile-friendly user interface on a web project (Bootstrap 2011). It has a grid system which provides an advantage by aligning content on different devices responsively. Bootstrap also offers various UI components, which helps the application look more desirable in the mobile version.

Application Flowchart

An application flowchart is one type of diagram that presents a simple workflow or process (Frank 1921). A good application flowchart can help the decision-maker understand what the application process will be. Therefore, they can plan before the project moves

forward.

Three flowcharts present the application process for DiSpend. Those application processes are: user registration and login, application processing, and data visualization. Each of them includes some flowchart keys, the combination of tables, to show the steps of application process.

Flowchart Keys

In figure 1, the flowchart key describes how the process steps working on DiSpend. When the application functions are located at the same page, which means they list in the same application layer, the tables will display as a same color. Also, the numbers appear on the top-right corner showing the table's relationship between the contents and its sub-contents. For example, in figure 1, when the user click the signup button (2.0) in the content page, DiSpend will go to another content page which has email address (2.1), password (2.2), and retype password (2.3) fields to fill in.

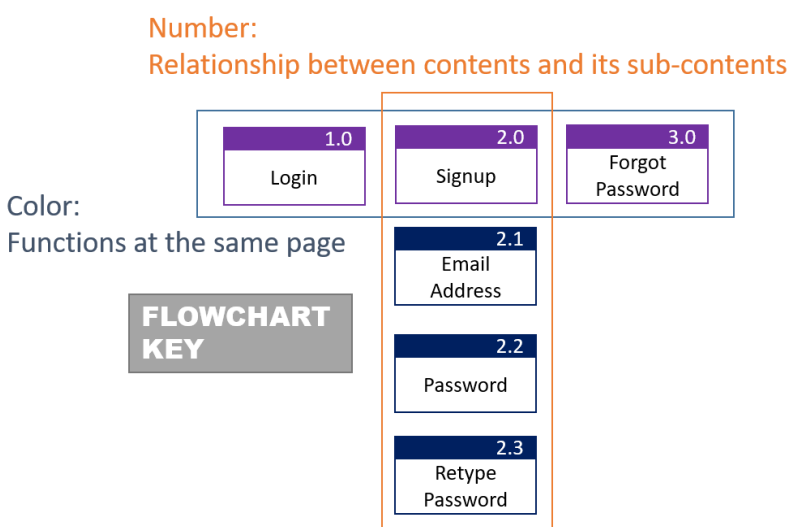


Figure 1. Flowchart key illustration.

After understanding how the flowchart keys work, there are flowcharts mentioned as

follow.

Flowchart 1 - User registration and login

Registration and login functions allow the user to use an app. These functions keep information private. At first, DiSpend requires the user first to sign up and then log in to the application. The operating flows should consider simplicity and security methodologies. When the user signs up, the process should be as simple as possible. The user should only fill in fields necessary to provide access to the app.

For simplicity, DiSpend requires only two fields, the user's email and a password allow the user to signup for the application. The less time users spend on registration, the easier it is for users to become members of this application. For security, sending a generated link to users can block the spam or junk emails. When the user clicks this generated link in their mailbox, it will validate the user's email into application system to confirm as an active account. The link should be time-limited in order to prevent other people from stealing the link. It will expire after 24 hours, and the user must type their email and password to register again. (see figure 2)

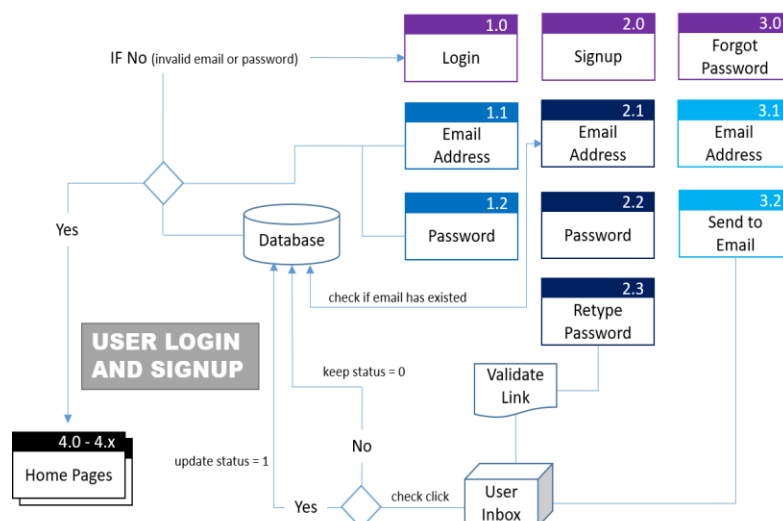


Figure 2. Application flowchart for user login and sign up.

Flowchart 2 - Application Processing

In figure 3, after the user logs into DiSpend, the main page should display three statuses for time: current task, upcoming task, and past task. The current tasks are the activities users are engaged in at the present time. The upcoming tasks are those the user will address in the future. The past tasks listed all previous work completed by the user.

The order in which the tasks appear, is the order in which the user will most likely address those tasks. The user can click this area to edit existing tasks or add new tasks. It includes how much time users spent on their tasks. Besides, DiSpend provides the statuses of the task. The user can choose succeed, failed, or in process for the task. (see figure 3)

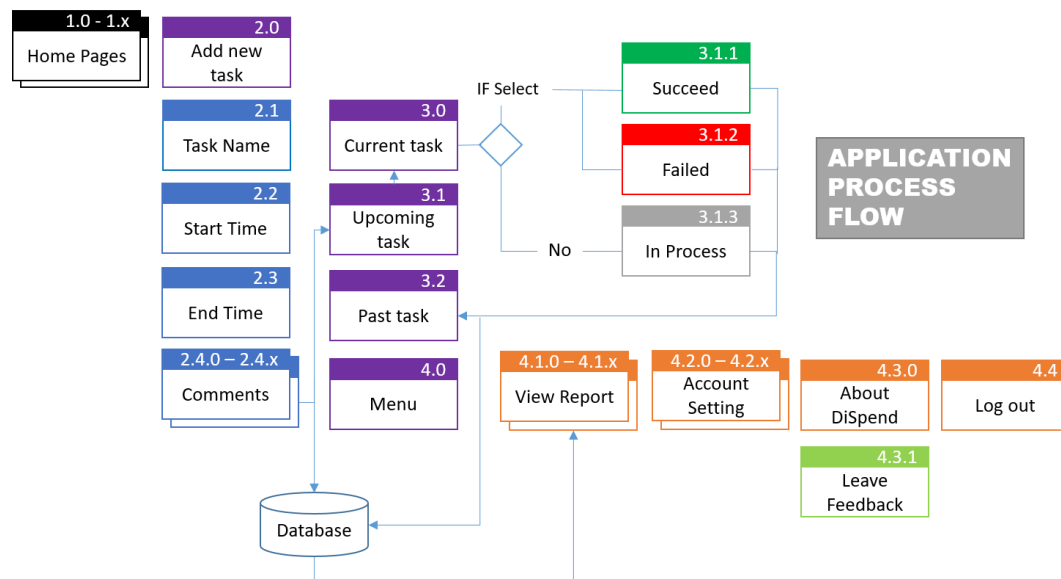


Figure 3. Application process flow for the main page.

Flowchart 3 - Data Visualization

Data visualization in DiSpend means the application presents a graphical version of the time spent by the user on a particular task. The user first clicks a button to select a time period, such as day, week, or month. Then the user clicks the report function on the home page, the database will gather the amount of time that require for the given task. After using the particular web API, like Chart.js, the time spent can be analyzed as a pie chart, a bar chart or a line chart to let the user view the report. (see figure 4 and figure 10)

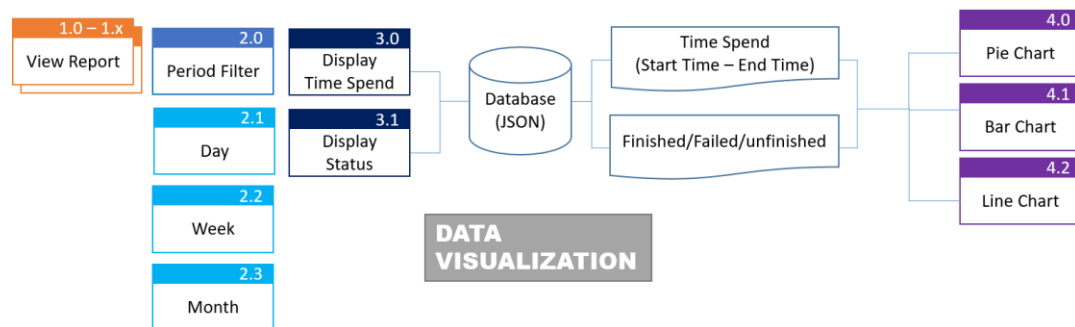


Figure 4. Data visualization flowchart for viewing the report.

Unified Modeling Language (UML)

Unified Modeling Language (UML) is a development tool that provides different methodologies that help the developer visualize the system design. In 2005, the Object Management Group's documentations defined UML 2.0 as thirteen types of diagrams (OMG, Inc. 2005). When the designer uses an appropriate UML diagram, the developer can have a better understanding of development, and quickly import the code module into their application for business functionality.

DiSpend contains two types of diagrams, which are *use case diagram* and *sequence*

diagram, and they are the most common UMLs in the world. Both are helpful for developing prototyping models.

UML 1 - Use Case Diagram

A system use case diagram describes a sequence of actions. The use case diagram uses circles to signify the relationship between each actor. An actor can be a person, organization, or even an external system that interacts with the application (Scott 2003).

DiSpend can apply use case diagram to let the developer understand the process from a user to become a member. In figure 5, the actors include a user and a member. A user has to complete the registration, validation, and authorization processes through DiSpend to become a member. Compared with a user, a member has more capabilities they can engage in such as setting up an account, tracking time, viewing reports, and receiving feedback from the application. Some information such as user account, time spent, and task status have to be stored in a database to have better control and management. (see figure 5)

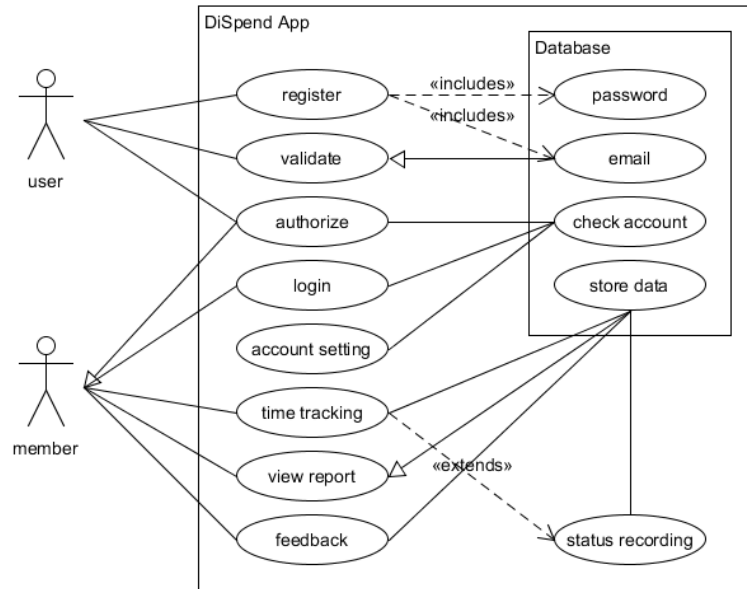


Figure 5. Use Case diagram.

UML 2 - Sequence Diagram

A sequence diagram is used to describe how the objects interact with other objects in a given situation (Inghelbrecht 2012). These objects include the following: the user, application, and database. A sequence diagram includes parallel vertical lines and different processes that happen simultaneously. The sequence diagram also includes arrows that signify the messages that are exchanged between the objects. The figure 6 below describes three objects: user, application, and database connecting with three vertical lines corresponding with each process.

The sequence diagram aligns the relationship between the user, application and the database. Also, it shows the processes of registration, login, main page and logout that happening in DiSpend. (see figure 6)

In the registration process, when the user fills in the form, the DiSpend application will insert the user's email and password into the database. At the same time, the database will generate a random link via the application to send a validated link to the user mailbox. When

the user clicks the link, it will update a new status in the database and confirms the user as a member.

In the login process, after the database establishes the user as a member, the database will respond to the application to allow the user to enter into the main page.

In the main page, when the user starts using the application, every action the user completes is going to update the database. It is because the data should be stored permanently to use in the future.

In the logout process, the application ends the session and forces the user back to the login page. This prevents the user from accessing the database for security improvement.

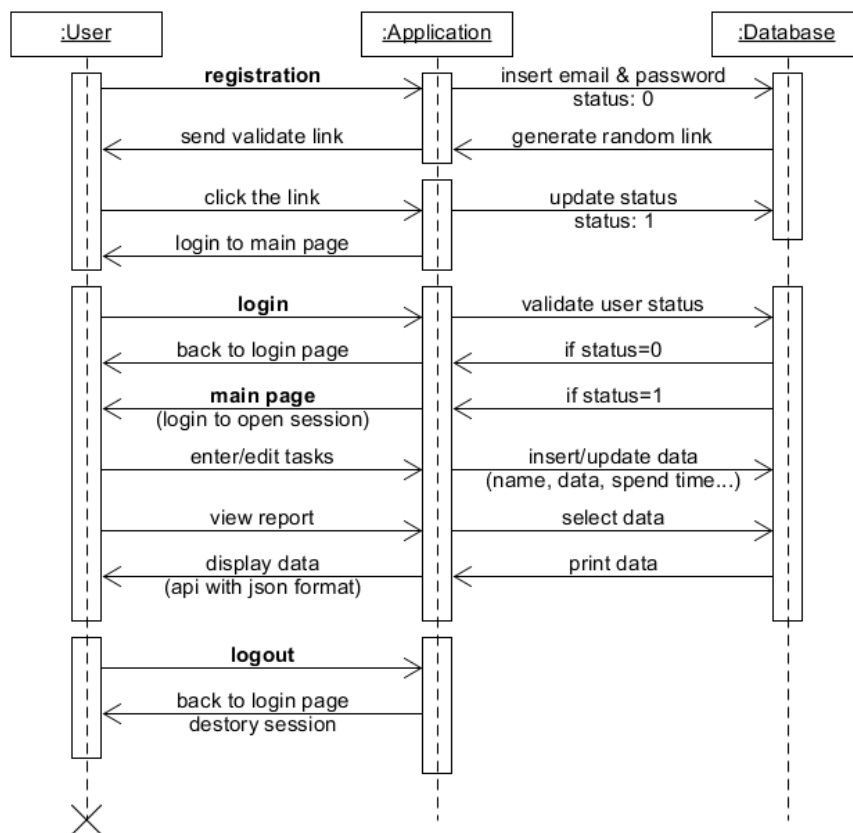


Figure 6. Sequence diagram for describing the relationship between the user, application, and the database. Also, for the process of registration, login, main page and logout.

Prototyping Model

After completing the planning designs for DiSpend application, the process can move to create a real prototyping model.

Prototyping is a design model that helps with system development (Margaret 2005). When the designer builds the prototype, the user is actively involved in the development. This process allows the developer to step the user through a prototype to detect errors. The developer can also identify the complex functions that might confuse other users (ISTQB 2013). In other words, prototyping collects user feedback to lead a better solution before the project is published and executed. The following are four functions created that describe how the DiSpend application works and deals with the time-tracking process.

Function 1 - Splash Screen

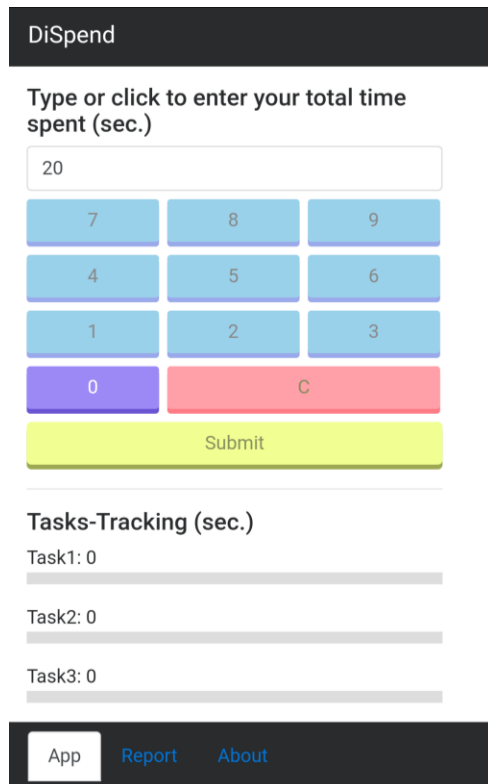
When the user wishes to use DiSpend, a splash screen is a welcome page that captures the user's attention (Computer Hope 2017). The advantage of splash screen is that the developer can create effects or provide information when the user visits the site for the first time. The splash screen indicates whether or not the application is working properly, and offers a basic introduction before using this app. (see figure 7)



Figure 7. Splash screen.

Function 2 - Number Keyboard

DiSpend is a time-tracking application. It means users should enter a time they expect to spend and the application will perform the calculation after users enter their time spent. The time can be a second, or a minute, or even an hour, but it must be a number. DiSpend creates a number keyboard to let the user enter the expected time the user plans to spend on the task. Through this functionality, the user can just press buttons to input their time. They can also press Clear - C or Reset button to add a new value. (see figure 8)



The screenshot shows the DiSpend application interface. At the top, there is a dark header with the text "DiSpend". Below the header, the instruction "Type or click to enter your total time spent (sec.)" is displayed. A text input field contains the number "20". Below the input field is a numeric keypad with buttons for digits 0-9, a clear button labeled "C", and a yellow "Submit" button. Below the keypad, the section "Tasks-Tracking (sec.)" is shown, containing three rows: "Task1: 0", "Task2: 0", and "Task3: 0", each followed by a horizontal progress bar. At the bottom, a dark footer contains three links: "App", "Report", and "About".

Figure 8. Number keyboard.

Function 3 - Progress Bar

If the application only displays time spent, the user may not know how much time they have left on the task. As a result, DiSpend designs a progress bar to show each task status. A progress bar is a graphical control element that used to visualize the process and expressed as a percentage (Mitchell 1979). By viewing the progress bar, DiSpend's user could have a better understanding of the time they spend and the time they left before this task is being over. The progress bar uses Bootstrap framework and combines with CSS, HTML, and JavaScript to interact with users. (see figure 9)

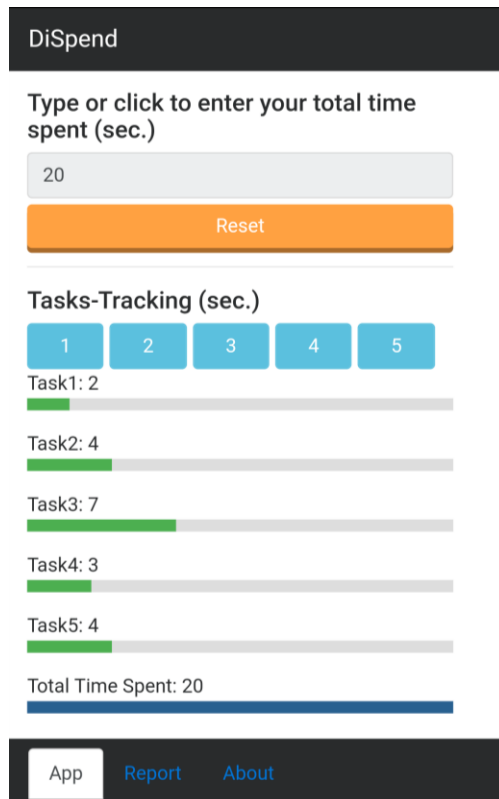


Figure 9. Progress bar.

Function 4 - Chart Report

After the application collects time spent and stores the data into the database, DiSpend will analyze the collection and display it as a chart report. The chart report is using a JSON format to exchange data. JSON can be viewed as a text, which has been widely used by developers to convert data as a JavaScript object (W3C 1999). In DiSpend application, after the time spent data is converted to a JavaScript object, it can integrate with Chart.js. Chart.js is an open source JavaScript charting for designers and developers. Chart.js simply loads the JSON data and provides a clear visual distinction between datasets in a chart format as shown in figure 10. The user can through this report measure how much time they may need next time for that task. (see figure 10)

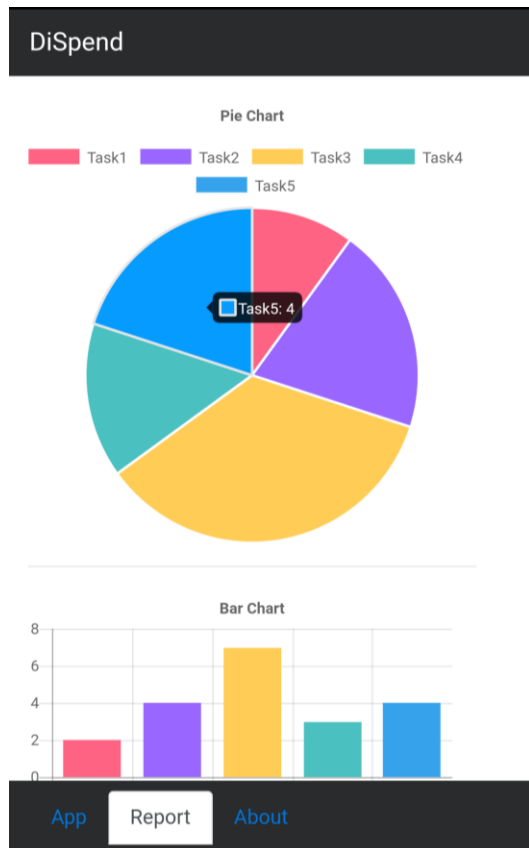


Figure 10. Chart report.

Finally, when DiSpend is designed and developed, it experienced some problems. Most of the designs could not be fully implemented because of a lack of full-stack knowledge and the development time is limited. The prototyping encountered a real-time tracking issue – the app could not count accurate in the device background when running other applications.

These problems make me rethink what is the best way to build my mobile app. Then I found that three methods that have been widely used for now, which are native app, hybrid app, and HTML5 app.

Analysis of the Proper Use of Native App, Hybrid App, and HTML5 App Methodologies

Today, mobile devices have become more important in our life. In a ComScore report *Number of Global User*, the number of mobile users exceeded the number of desktop users in 2013 (ComScore 2016). Additionally, the ComScore report shows that mobile technologies have overtaken desktop technology based upon the number of users. (see figure 11)

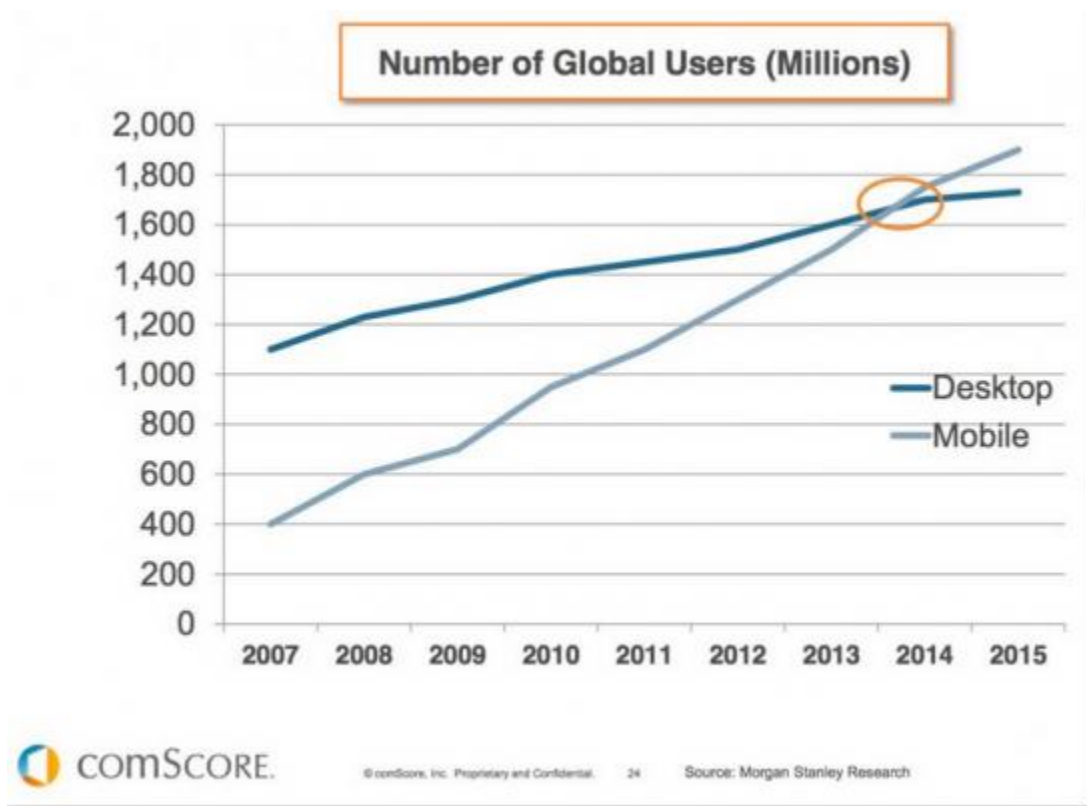


Figure 11. ComScore report, Number of global users.

Source: Morgan Stanley Research. "In 2015 the number of global mobile users exceeded the number of global desktops." Accessed 2016. <https://www.comscore.com/Insights/Presentations-and-Whitepapers/2016/The-2016-US-Mobile-App-Report>

However, even as mobile technologies have already entered into the world, many enterprises introduced mobile application development without truly thinking of the difference among each approach (Stangarone 2016). The three mainstream approaches for developing mobile applications include the following: native apps, HTML5 apps, and hybrid apps.

Native apps, HTML5 apps, and hybrid apps are widely used for mobile development.

This analysis section aims to compare the advantages and the disadvantages of the native app, HTML5 app, and hybrid app approaches. The purpose is to help companies understand what is the best approach for mobile application development.

Native App Development

Native apps are the universal mobile application that people used to download from the App Store. Native apps are written for a particular platform. The current common platforms are iOS, Android, and Windows. These specific platforms require different development methods.

iOS platform

In the iOS platform, the developers have to set up their coding environment with Xcode, Swift, and the iOS SDK. They use Apple's Integrated Development Environment (IDE) to build for both Mac and iOS apps. Xcode is a graphical interface that assists developers in writing their applications. The developers use Swift programming language, which is developed by Apple, written on the Xcode platform to be used in an iOS app. Xcode also works well with Objective-C. However, Swift is only available for the Mac, so if developers want to make iOS apps, they must use a device that runs OS X, which is an operating system built upon a Unix core and resides on Apple's desktop and a portable computer (OSX 2017). After the application is finished by the developer, the developer should follow the Apple App Store Review Guidelines before the application is published. There are some limitations presented on Apple's App Store Review Guidelines such as safety, performance, business, design, and legal that have to be understood and addressed before the application is published (Apple's Guideline 2017). The developers are

required to pay 99 dollars a year for an annual fee after their apps upload on the Apple Apps Store (Klosowski 2014).

Android platform

In the Android platform, the developers can download the Android Studio to set up the development environment. Android Studio is the most common IDE for Android development, which comes from Google, Inc. Android Studio provides a UI to let developers enter their code, it also offers a selection of tools called the Android Software Development Kit (Android SDK), and an emulator that renders a virtual device to test the apps and it is referred to as the Android Virtual Device. Android Studio is convenient for developers by using the existing code provided by Android. Android also provides rules on how to build an Android app. Eclipse is another popular IDE that uses Java for Android development. It is free for Windows, Mac, and Linux operating systems. After choosing the preferred IDE, the developers can follow the official *Android Developer Guidelines* and *Android Design Guidelines* to start a new project. When the app is done, Google Play has a one-time publishing fee of \$25 (Ravenscraft 2014).

Windows platform

In the Windows platform, according to the Microsoft Windows Dev Center (year) documentation, the developer can download Windows Phone SDK to create a new project in Windows 8. For the Windows 10 version, Microsoft has a new release of the Universal Windows Platform (UWP). UWP is flexible platform and allows the developer to create their project either with the Design UI or write C# and XAML codes. UWP also has many features such as API integration, in-app purchasing, and report analysis (Windows Dev Center 2017).

Hybrid App Development

A hybrid app is a native app wrapper containing a browser that accesses a single mobile web app. (Bristowe 2015) Hybrid app uses Web views, which is a mobile platform that is hosted inside a native container, to build with in combination with web technologies like HTML, CSS, and JavaScript. In other words, Hybrid app uses a container to wrap HTML5 app to make it possible access all the mobile features as native. Hybrid app uses web programming that is based on a mobile development framework distributed on the App Store. It is widely used for apps that do not have high-performance requirements but needs full device access as a native. For example, apps that only need to provide information, or to upload images can be built quickly by implementing the hybrid app approach.

Apache Cordova

Today, most of the hybrid apps leverage Apache Cordova. Cordova allows a developer to use their existing skills in web development for mobile app development. Cordova command-line runs well on Node.js and is available on Node Package Manager (NPM), which is a package manager for JavaScript (Node.js 2017). The developers can easily write one set of code to target mobile or tablet devices to publish a single app for all devices on the different app stores. This allows the developer to build a mobile application with almost no difference as the native app.

HTML5 App Development

HTML5 app, which is called the Mobile Web, also uses web programming like the hybrid app but the difference is HTML5 apps are distributed on the web. HTML5 is the newest version of Hypertext Markup Language used to create web pages (Gary 2011). HTML5 has new features

that mobile device can access. Native behaviors such as push notification, detect foreground, sync background, store offline, control media, access geolocation is available with HTML5. It has standardization that lets the developers work seamlessly on different browsers so that they can easily build an application. When building an HTML5 app, there are two techniques to consider for mobile development, which are Responsive Web Design and Progressive Web App.

Responsive Web Design

Due to HTML5 apps are responsive mobile web applications that can be directly viewed via a browser, Responsive Web Design (RWD) is one method that commonly used to make the application automatically fit on different devices. When users use their desktop, laptop, or mobile device via a browser, the application responds to align appropriate content based on their screen size, platform, and orientation (The Smashing Team 2011).

Progressive Web App

Progressive Web App (PWA) is another technique. It was an advanced approach for developing web apps announced by Google at the 2015 Web Summit Conference. PWA lets an application able to push notifications and access content offline among other features similar to Native via the web browser. In addition, there will be other features capabilities for delivering an app that acts as a native app user experience. Such as adding an icon to the home screen, launching in full-screen with URL bar or displaying a splash screen effect (Google APIs 2014).

How to choose a right approach?

Since there have many ways to build mobile applications, the considerations for the enterprise to select an appropriate approach becoming important. When the company decides to choose the right approach, different perspectives such as performance, speed, cost, and security need to be considered.

Performance, Speed, and Cost

Native Performance

According to the Mobile App Comparison Chart (see table 1), Native apps lead on performance over hybrid apps and HTML5 apps. Native apps are suitable for building a consumer-focused application or constructing a game. It uses built-in device components, such as a camera, a microphone, the contact book, geolocation and other features. Native apps also have different mobile categories such as iOS, Android and Windows phone can be programmed individually for each separate platform. For example, an iOS app needs to be deployed using an iOS SDK (Software Development Kit) platform. An Android app requires deployment using an Android SDK platform (Stangarone 2016).

Native apps, which are implemented using Objective-C, Java and other programming languages and are distributed on the App Store, are commonly used in phones. Because native apps have all the compatible features that can be accessed on a mobile device, it is suitable for constructing games or consumer-focused applications, which requires a heavy graphics load and high performance (Stangarone 2016).

Table 1. The Mobile App Comparison Chart: Hybrid vs. Native vs. Mobile Web.

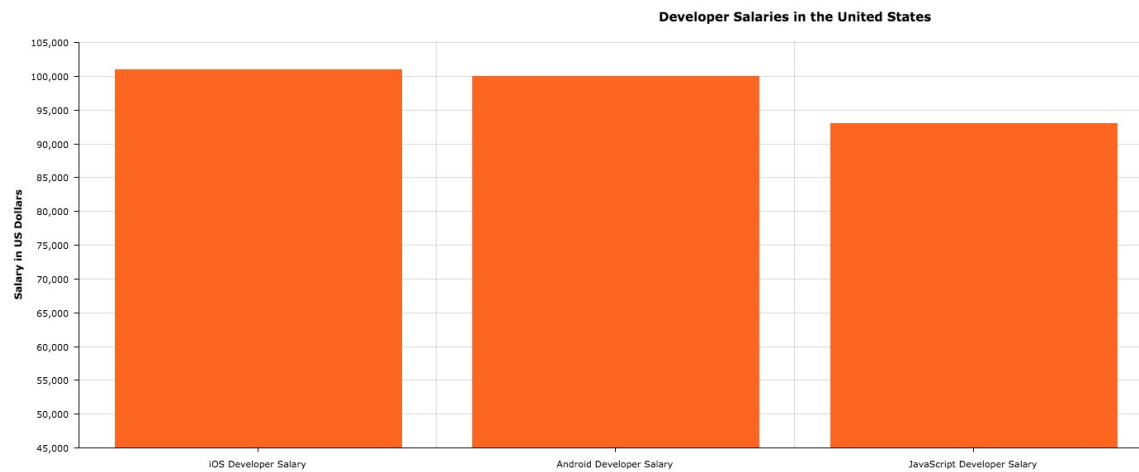
	Native	Hybrid	HTML5 (mobile web)
Skills needed to reach Android and iOS	Objective-C, iOS SDK, Java, Android SDK	HTML, CSS, Javascript, Mobile Development Framework	HTML, CSS, Javascript
Distribution	App Store/Market	App Store/Market	Web
Development speed	Slow (More Info)	Moderate	Fast
Development cost	High (More Info)	Moderate	Low
Maintenance cost	High (More Info)	Moderate	Low
Graphical performance	High	Moderate	Moderate
App performance	Fast	Moderate	Moderate

Source: Stangarone, Joe. "The Mobile App Comparison Chart: Hybrid vs. Native vs. Mobile Web." Mrc's Cup of Joe Blog. Published June 14, 2016. <http://www.mrc-productivity.com/blog/2016/06/the-mobile-app-comparison-chart-hybrid-vs-native-vs-mobile-web>

Native Cost and Development Speed

According to PayScale's developer salaries report, (see table 2) Native apps have a high development cost. It has become increasingly difficult for new startups who do not have enough funds to create native apps to develop a minimum viable product (MVPs) and prototypes (Nader 2016). That also means the developers have to spend more time to write and maintain the code for each operating system and for devices that require unique code for the app to work on the device.

Table 2. Developer Salaries in the United States.



Source: PayScale. "Average cost of employing iOS, Android, and JavaScript developers in the United States." Updated 2017. <http://www.payscale.com/research/US/Skill=JavaScript/Salary>

HTML5 and Hybrid Advantages

Although hybrid app and HTML5 app have average performance, they are faster to develop than native app. In figure 12 shows that the time needed to develop a basic type of hybrid app or mobile web app (HTML5 apps) is approximate about 1 to 3 months, which is half of the development time for a native app (YourCloud 2015).

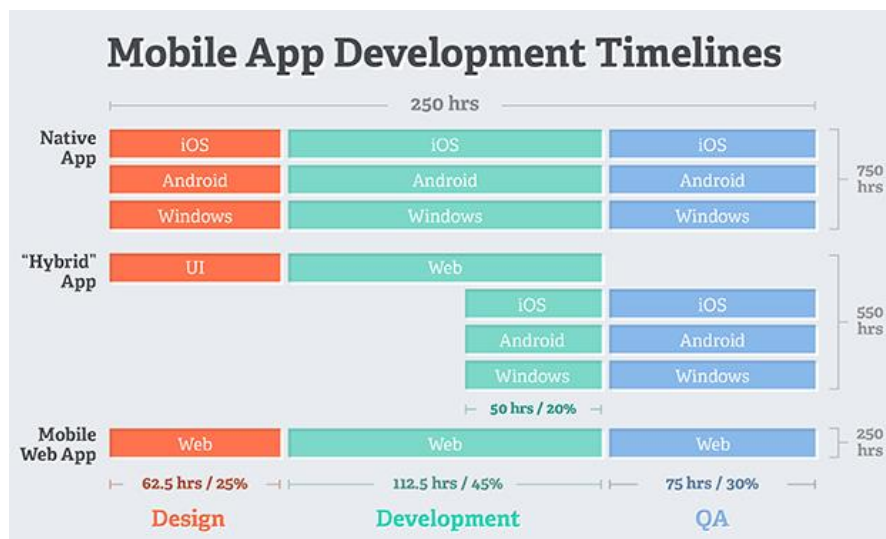


Figure 12. Mobile app development estimated timelines.

Source: YourCloud, 2015. "Mobile app development estimated timelines." Hussain Sabib posted Quora on October

10, 2016. Accessed 2015. <https://www.quora.com/What-is-the-average-development-time-for-an-iPhone-app>

HTML5 apps have the fastest development speed and lowest cost features. HTML5 app can be built rapidly by the developer with web programming and is low-cost development method for the enterprise. However, since HTML5 apps, mobile device access has come a long way, HTML5 apps may be limited by the features and functions it can control. For example, in the “What Web Can Do Today” website, HTML5 still lacks some common mobile features, such as open Bluetooth, turn on NFC, detect network type and speed, unlock the wake lock, and access phone contact information. Those limitations are device dependent or still in the experimental stage (Adam 2017).

Hybrid apps are the best solution for the developer to build an application that does not have high-performance requirements but needs full mobile device access. Popular frameworks such as React Native, which is published by Facebook, tried to bridge the gap between the performance of native apps with the ease of development of HTML5 apps. React Native supports both iOS and Android operating systems. It uses open source library, reuses the native components and renders mobile UIs so the developers can build apps with more agile approaches. React Native also offers third-party plugins compatibility without relying on WebView. For example, if the developers want to embed Google Map API into the application, React Native lets them link the plugin with a native module, so the map can up with the mobile device’s functions like rotate, zoom and compass. Integrate plugins has less memory usage and fast loading speed advantages, to make the hybrid apps have better performance as native (Carey 2016).

The Future of Mobile Development

Hybrid app accounts for both the native app and HTML5 app pros and cons to reach a compromise. As web development technology keeps improving, HTML5 apps should be expected to control more features and functions on mobile devices. In general, Native apps are costly but provide the best features, the best usability, and the best mobile experience with respect to performance. But now, the advanced React Native framework should be the leading of hybrid app development in next few years (Carey 2016).

Security Overviews

Currently, our mobile devices allow us to do nearly everything online. Although the transformation of mobile communications brought enterprises additional strategies to grow their revenue, improve their profit, and increase productivity, only a few of them are concerned about security risks (Shaukat 2016). Arxan Technology's report, *Application Security Report Reveals Disparity between Mobile App Security Perception and Reality*, shows that of the 90 percent of apps surveyed, two out of ten apps had a significant security risk as defined by the Open Web Application Security Project (OWASP). Hackers can steal, intercept, and access your phone for any purpose. As a result, when mobile app development companies consider where to spend money, it is better to secure the mobile app by protecting the code, checking the network, applying authorization, and other security methods (Carey 2016).

Code Protection

To protect the code, developers write the app code with encryption. It is better to keep the code to be secret, as well as hard to read. There are some algorithms that can be used to

prevent a cyber-attack. For example, Triple DES encryption protects data on three individual keys with 56 bits on each key, which is a symmetric-key block cipher that widely applies to financial services and other industries (Coppersmith 1996). RSA is another encryption. RSA stands for Ron Rivest, Adi Shamir and Leonard Adleman, who first published in 1978. RSA uses a public key to encrypt a message and uses a private key to decrypt it. This asymmetric algorithm takes the attackers spending more time to hack into the data (Rahman 2012).

Network Security

The common solution for network security is to connect using a virtual private network (VPN), applying SSL (secure sockets layer) or TLS (transport layer security). These verified Internet protocols protect the sensitive information when passing the data from the client back to the application's server and database (Carey 2016).

In those Internet protocols, SSL encryption is the most common transport security type that is widely used for mobile. The Hewlett Packard report *Mobile Application Security Study* shows that eighteen percent of applications sent their usernames and passwords over HTTP, while another 18% implemented SSL/HTTPS incorrectly (Hewlett Packard Report 2013). When the user connects to the Internet, that connection passes through the external firewall to a server using SSL if the application is applying HTTPS protocol. The HTTPS encrypted the data transmission and kept the user data as private on the Internet (Stephen 2000).

The current reality is most mobile applications connect to the web to share information with users around the world. In VisionMobile's survey, *Developer Economics 2013*, over 47 percent of all iOS apps and 42 percent of Android apps rely on HTML5, showing the importance of "web" technology is rapidly growing in mobile (VisionMobile Report 2013). As a result,

integrating SSL into company mobile development benefits the application of transport security.

Encrypt Authorization

JSON Web Tokens (JWT) is a standard protocol that used to encrypt a JSON object when applying authorization. It encrypts data by using RSA or HMAC (Hash-based message authentication code) algorithm and ideal for mobile security (Auth0 2013). OpenID Connect is another federation protocol that specifically designed for mobile. It allows developers to authenticate their users to reuse the same credentials across the web to apps with an ID token, so they do not have to register or sign-in at each time (Carey 2016). Both provide a safe way to store the user data.

When security becomes the vital part of the mobile device, an insecure data storage will leave the enterprise to lose their private data. Research conducted by Hewlett Packard(HP), describes that 75% of applications do not use proper encryption technique when storing the user data on a mobile device (Hewlett Packard Report 2013). For example, most of HTML5 apps require a one-time password authentication, which saves the user identity when they login to increase a better user experience. However, by saving the user's identity, this presents an opportunity for hackers to steal that identity. On the other hand, native app also keeps the user identity. Instead of exposing the user information on the public Internet, native app aids security by using built-in mobile components that are encrypted storage into the device itself (Carey 2016).

Security Architectures

No matter which ways companies decide to secure the mobile app, to prevent company information from being stolen by hackers, several aspects such as confidentiality, integrity, authentication, authorization, availability, and non-repudiation should be asked for before following the lifecycle of mobile security architecture (Shauka 2016). For confidentiality, enterprises should regularly think about how to keep their application data private. For integrity, organizations should examine the movement of data from outsourcing APIs to make sure they are trusted and verified. For authentication, check the application to verify all the user's identity correctly to prevent spam, anonymous mobile registration. For authorization, it is important to use the proper app to limit some users' privileges. For availability, think about the different ways that hackers may harm the mobile application. For non-repudiation, understand the application parts that record and track the user's data and information (Shauka 2016). By putting these questions into approaches, the security can be considerate comprehensive while developing a new app.

The Final Decision among Hybrid or Native or HTML5

After viewing the different perspectives from performance, cost, speed and security. A company can decide which is the best approach for their mobile application development method.

For performance, cost and speed, native app has its advantages for the best usability, full access to hardware API, and easy to integrate with other native elements. HTML5 app has its advantage with the fastest development speed, low-cost benefits, cross platform portability,

and large open source communities. A hybrid app has the moderate levels between both of them, so the company can choose it if they want to build a native application but when the budget is limited.

For security, when it comes to native app, insecure local data storage, unintended data leaks, and other code injection might be the biggest problems. When it comes to hybrid app, JavaScript injection, weak SSL implementation, and other caching issues stand out to the stage (Jagtap 2015). However, native app currently is still regarded as more secure than hybrid and HTML5 app (Thomas 2014).

In conclusion, if the company wants to develop an app such as banking or e-commerce that requires high performance and security, it is better to use a native app. If the company just wants an app that provides non-sensitive information to their user, then the hybrid app or HTML5 app will save cost and time. No matter what, developers should follow the secure coding guidelines during the process, and regularly conduct intervals on penetration testing after finish the application.

DiSpend time-tracking application is using HTML5 app to build. It is because HTML5 app has the agile development speed. Besides, DiSpend does not require much security factors as it is a prototyping demo.

Reflection: Mobile Application Development

Why I chose this project?

Foreword

The story started when I worked in a web planning position in my country. I was helping the company upload some products for the public on its E-commerce website. While doing these routine tasks, I found I was wasting time with each upload process. That pushed me to become a web developer because as a developer, I could make this entire process much easier. For example, by making a time management application to record different working tasks, my work could be effective and efficient.

I decided to go abroad to pursue my master's degree; I went to Academy of Art University to study web design at first because I want to learn some user interface (UI) and user experience (UX) strategies. After that, I transferred to the Web Development and Web Design program at the University of Denver's University College. After I had enrolled in a few classes, I realized that the Web Development courses could not satisfy my curiosity, because some of the courses were too similar to my previous education and work. I switched my concentration to Mobile Application Development in the Information Communication Technology (ICT) program. Now, I am taking Creative Capstone and working part-time on campus as a website developer.

I found that Mobile Application Development program is designed for someone who wants to learn more advanced technology on mobile devices, and it is not limited to web development. I felt that I was at the right place. I have both design and development in my background, and all I need is to enhance those skills.

After taking several mobile application courses, I felt that I lacked how to implement a

project. That is the reason why I decided to choose Creative Capstone Project for my Capstone. The Creative Capstone can demonstrate that I can build an application. In addition, I can use this application to track the time spent to be organized.

What I have learned from University of Denver and used for this project

The following are the skills that I learned from the Mobile Application Development program, which I can add more value to current position and applied to this project.

Mobile Design

Mobile Design is different from the web. Because the mobile version has a smaller screen, the content should be designed as simple as possible. For example, I can apply Bootstrap, an HTML5, CSS, and JavaScript framework to make the content of columns align appropriately and provide a better user experience. In ICT-4510 Advanced Web Design and Management, the course covered responsive web design concepts to make the website look mobile-friendly. In DiSpend project, I learned how to make the font-size much bigger, adjust the image to fit 100% of the mobile device, and use SVG technique to design small graphics or icons to solve display issues.

Coding Skills

ICT-4510 Advanced Web Design and Management course reviewed my existing HTML5, CSS3, and JavaScript programming skills. It also guided me on how to integrate outsourcing Application Program Interface (API), like Google Map API and Flickr API, using Ajax, and JSON syntax to write object-oriented codes. ICT-4561 Web Development with PHP course taught me

how to set up a LAMP (Linux/Apache/MySQL/PHP) environment and effectively use PHP to construct a proper application. For instance, the course taught me using PHP to handle form data, to upload files, and to connect with MySQL database. In DiSpend project, even I did not implement the database connection, but it give a basic concept of how to design a complete chart flow, such as creating the user registration and login process.

Mobile Application Development

In ICT-4580 Mobile Application Development course, covered Apache Cordova. Cordova is a platform that allows web developers to use existing skills such as HTML5, CSS3, and JavaScript to write a hybrid application. In DiSpend, I used Cordova to build a hybrid application. Cordova lets me enhance my web development skills and do not have to learn new programmings such as Java or C# or something.

Design Modeling

I enrolled in ICT-4300 Web Enabled Info Systems. The course introduced the basic web design and programming principles, such as object-oriented modeling (OOM) methods, use case development, class and sequence diagrams. The course covered software development methodologies such as Waterfall, Prototyping, Agile, Scrum, and Extreme Programming (XP). The course included some hands-on lab exercises using Java Software Development tools (JDK) with jGrasp and Database Management System with MySQL. In DiSpend, I used the OOM methods that I learned from this course to create multiple diagrams for a prototyping.

Information Security

There are many websites such as banking or e-commerce that require encrypted information systems for security. Cryptography is one of the eight domains required for the Certified Information Systems Security Professional (CISSP). ICT-4680 Principles of Cryptography is one of the courses that provided cryptographic techniques that are used to protect a company's sensitive data. Different topics include Advanced Encryption Standard (AES), the Secure Hash Algorithm (SHA), Digital Signatures Authentication, public-key infrastructure (PKI), secure sockets layer (SSL), and more. The Cryptography course gave me an idea on how to analyze and compare the security issues among the native apps, hybrid apps, and HTML5 apps.

Business and Information Technology initiatives

University College provides three individual foundation courses which are Business Foundation, Technical Foundation, and Enterprise Architecture. Business Foundation provides an overview of the relationships between business and ICT solutions. Technical Foundation reviews the technology at the core of the ICT industry. Enterprise Architecture uses the enterprise-wide frameworks to integrate business initiatives into ICT program. Enterprise Architecture course is useful on DiSpend because it let me think of applying mobile technologies into business consideration.

Skills and Knowledge Demonstrate

DiSpend is a mobile application for tracking the user's time spent. Even with prototyping, certain technologies are involved throughout the project. After introduced what I have learned from the University of Denver, DiSpend demonstrates the application of the following skills and knowledge acquire in the program:

Agile Modeling

I learned agile modeling skills when I first designed DiSpend. The application flowchart allows me clarify the requirements of the system. The prototyping model assisted me to understand what the final user interface and functions would be. The object-oriented design such as the Use Case diagram and the Sequence diagram benefited and accelerated the project implementation.

CSS3

When the latest version of the Cascading Style Sheets 3(CSS3) was released, it allowed developers to do more transitions and effects on the web. DiSpend applied CSS3 to mouse hover transition. For example, in DiSpend, while the user mouse would move to the number keyboard, the CSS3 would make the button display different styling colors interaction. The shadow effect made elements of the buttons look more vivid.

JavaScript

DiSpend used JavaScript programming because the application needs to perform the complex functions on web pages. For example, to run a timer, the application must set a new interval time by counting a number of seconds. To display a report, the application has to integrate with Chart.js, a JavaScript API that is used for showing dynamic charts. To receive the data, the application should transfer the user's time as a JSON format to do calculations.

Github

Github is a platform that allows the developer to control their application version and cooperate with other developers. That means when DiSpend is published on the Github, every

step of modification will be recorded. For example, if I deleted a line of codes accidentally, I could take a look at Github's history to recover my mistakes. I could also share my DiSpend app to other developers to let them work on my project to reduce the bugs and add more functions.

Web Hosting

When the application is published on the Internet, it must have an actual space where it can reside. The DiSpend prototype is stored on a shared web hosting site called Bluehost. Bluehost provides an online web hosting service and domain registration. So if the user is typing a certain URL on their browser, which is a unique domain name, the browser will request to the server, which is hosted on Bluehost, to display DiSpend to the user.

Cordova Android

Apart from the HTML5 web application, DiSpend also has another version for hybrid application. (see figure 12) DiSpend is a hybrid app that was built using Cordova; it is a platform that allows me to write an application and deploy it on a mobile device like a native app. I installed Cordova on Android Studio so that DiSpend could publish on my Android phone desktop. Compared to HTML5 app, this mobile app has more accessible features such as push notification, offline access, and run in background mode which allows the user to open other apps at the same time.

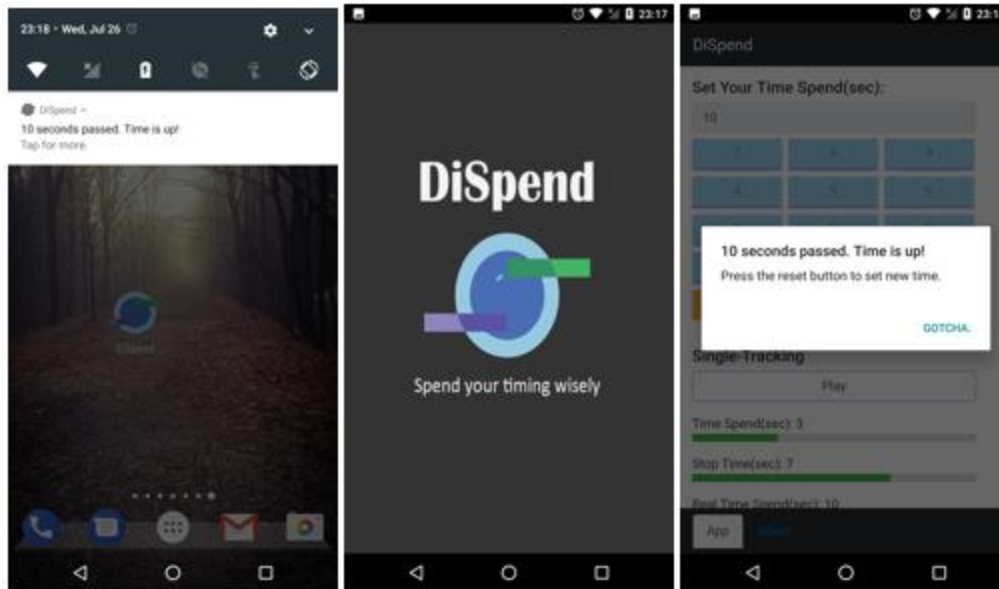


Figure 13. DiSpend hybrid application that built with Cordova.

Final Perspective

DiSpend application follows the full-stack design and development methods. By planning the design, I gained knowledge on how to create better usability, achieve a well-organized application flowchart, and draw a useful UML diagram. By implementing the development, based on four types of prototyping models: splash screen, number keyboard, progress bar, and chart report, I combined various technologies such as Agile, JavaScript, Github, Web Hosting, and Cordova Android in my project. These skills that I used on DiSpend allowed me to become more professional in the web and mobile development area. By analyzing the differences between the native app, hybrid app, and HTML5 app. I realized that there is not only one way to build a mobile application. When choosing the development methods, the decision should comprehensively consider the app qualities of performance, cost, speed and security. Finally, I was grateful that the University of Denver courses guided with mobile application development to finish this project. I believe DiSpend will become not only a

prototype but a functioning mobile application that can help people to manage their daily tasks and improve their self-discipline.

References

- Adam, Bar. 2017. "What Web Can Do Today." *An overview of the device integration HTML5 APIs. Can I rely on the Web Platform features to build my app?* Updated 2017. <https://whatwebcando.today/>
- Apple's Guidelines. 2017. "App Store Review Guidelines." *Apple, Inc., Apple Developer, App Store, App Review, App Store Review Guidelines.* Updated 2017. <https://developer.apple.com/app-store/review/guidelines/>
- Auth0, 2013. "Introduction to JSON Web Tokens." *What is JSON Web Token?* Accessed 2013-2017. <https://jwt.io/introduction/>
- Bootstrap. 2011. "Bootstrap." *Official website created by Otto, Mar, and Thornton, Jacob. Last modified 2017.* Accessed August 19, 2011. <http://getbootstrap.com/about/>
- Bristowe, John. 2015. "What is a Hybrid Mobile App?" *Progress Telerik Developer Network (mobile).* Accessed March 15, 2015. <http://developer.telerik.com/featured/what-is-a-hybrid-mobile-app/>
- Carey, Wodehouse. 2016.
- a. "7 Reasons Why Facebook's React Native Is the Future of Hybrid App Development." *Upwork Global Inc, Hiring Headquarters, Mobile Application.* Published January 13, 2016. <https://www.upwork.com/hiring/mobile/react-native-hybrid-app-development/>
 - b. "8 Tips for Better Mobile Application Security." *Upwork Global Inc, Hiring Headquarters, Mobile Application.* Published November 16, 2016. <https://www.upwork.com/hiring/mobile/mobile-application-security/>
- Computer Hope. 2017. "What is a Splash Screen?" *Free computer Help since 1998.* Updated April 26, 2017. <https://www.computerhope.com/jargon/s/splashsc.htm>

ComScore.com. 2016. "Number of Global Users." *ComScore U.S. Mobile App Report, Figure 4-1*.

Published 2016. <https://www.comscore.com/Insights/Presentations-and-Whitepapers/2016/The-2016-US-Mobile-App-Report>

Coppersmith, D., and Johnson, D.B. and Matyas, S. M. 1996. "A Proposed Mode for Triple-DES Encryption." *IBM Journal of Research and Development*. Published March 1996. Page 253.

<https://pdfs.semanticscholar.org/3d33/a6cf13d8fd7aef3086c7a3dcd77d295cace7.pdf>

Frank B., Gilbreth. 1921. "The American Society of Mechanical Engineers." *Process Charts*.

Published December 9, 1921. Page 4.

<https://engineering.purdue.edu/IE/GilbrethLibrary/gilbrethproject/processcharts.pdf>

Gary, Marshall. 2011. "HTML5: What is it?" *Techradar.com. The source for tech buying advice*. (News) Published December 13, 2011.

<http://www.techradar.com/news/internet/web/html5-what-is-it-1047393>

Gellert, Laurence. 2012. "What is a Full Stack developer?" *Laurence Gellert' Blog*. Published August 1, 2012. <http://www.laurencegellert.com/2012/08/what-is-a-full-stack-developer/>

Google APIs. 2014. "Progressive Web Apps." *Google Developers*. Last modified December 5, 2014. <https://developers.google.com/web/progressive-web-apps/>

Hewlett Packard Report. (HP) 2013. "Mobile Application Security Study" *Hewlett Packard Enterprise*. Accessed 2013-2017. <https://www.hpe.com/h20195/V2/GetPDF.aspx/4AA5-1057ENW.pdf>

Inghelbrecht, Yanic. 2012. "A Quick Introduction to UML Sequence Diagrams." *Trace Modeler*.

Published 2012.

[http://www.tracemodeler.com/articles/a quick introduction to uml sequence diagrams/](http://www.tracemodeler.com/articles/a-quick-introduction-to-uml-sequence-diagrams/)

ISTQB. 2013. "What is Prototype model - Advantages, Disadvantages and When to use it?"

ISTQB Exam Certification. Published 2013. <http://istqbexamcertification.com/what-is-prototype-model-advantages-disadvantages-and-when-to-use-it/>

Jagtap, Harshad. 2015. "Native v/s Hybrid Apps: Security Aspects." *Best Practices in Android Security, Defence*. Published June 11, 2015.

<http://www.appvigil.co/blog/2015/06/native-vs-hybrid-apps-security-aspects-of-each-of-the-apps/>

Klosowski, Thorin. 2014. "I Want to Write iOS Apps. Where Do I Start?" *Lifehacker.com*.

Published October 10, 2014. <http://lifehacker.com/i-want-to-write-ios-apps-where-do-i-start-1644802175>

Lewis, Edward. 2017. "ICT-4010 Enterprise Architecture" *Information Architecture*. Delivered Winter Quarter, 2017.

Linowski, Jakub. 2014. "A Good User Interface." *UI Designer at GoodUI*. Published May 2, 2014.

<http://www.goodui.org/>

Margaret, Rouse. 2005. "What is Prototype?" *TechTarget, Definition. Search Manufacturing ERP*. Updated August 5, 2005.

<http://searchmanufacturingerp.techtarget.com/definition/prototype>

- Mitchell L, Model. 1979. "Monitoring System Behavior in a Complex Computational Environment." *Department of Computer Science, Stanford University*. Published January, 1979. <https://searchworks.stanford.edu/view/762873>
- Nader, Dabit. 2016. "The Cost of Native Mobile App Development is Too Damn High!" *Software Consultant Specializing in Teaching and Building React Native, Hackernoon*. Published December 14, 2016. <https://hackernoon.com/the-cost-of-native-mobile-app-development-is-too-damn-high-4d258025033a>
- Node.js. 2017. "Node.js Foundation" *Joyent Company*. Updated 2017. <https://nodejs.org/en/>
- Object Management Group, Inc.(OMG) 2005. "Unified MOdeling Language: Infrastructure." *Language Architecture, Design Principles*. Publish May 7, 2005, Page 11. <http://www.omg.org/spec/UML/2.0/>
- OSX Daily, 2017. "Mac OS X Introduction." Accessed 2006-2017. <http://osxdaily.com/category/mac-os-x/>
- Rahman, Md et al., 2012. "Implementation of RSA Algorithm for Speech Data Encryption and Decryption." *IJCSNS International Journal of Computer Science and Network Security*, VOL.12 No.3. Published March 2012. http://paper.ijcsns.org/07_book/201203/20120312.pdf
- Ravenscraft, Eric. 2014. "I Want to Write Android Apps. Where Do I Start?" *Lifehacker.com*. Published October 8, 2014. <http://lifehacker.com/i-want-to-write-android-apps-where-do-i-start-1643818268>
- Scott W., Ambler. 2003. "UML 2 Use Case Diagrams: An Agile Introduction." Published 2003-2014. <http://www.agilemodeling.com/artifacts/useCaseDiagram.htm>

Shaukat, Humayun. 2016. "Mobile Security Testing." *Linkedin, Senior Consultant & SME Mobile Testing*. Accessed June 30, 2016. <https://www.linkedin.com/pulse/mobile-security-testing-humayun-shaukat>

Stangarone, Joe. 2016. "The Mobile App Comparison Chart: Hybrid vs. Native vs. Mobile Web." *The m-Power Development Platform. (Support, Tech Blog)* Published June 4, 2016. <http://www.mrc-productivity.com/blog/2016/06/the-mobile-app-comparison-chart-hybrid-vs-native-vs-mobile-web/>

Stephen, Thomas. 2000. "SSL And TLS Essentials." *Wiley Computer Publishing, Securing the Web*. Published 2000 by Robert, Ipsen. Edited by Marjorie, Spencer. Page 37. <https://cdn.preterhuman.net/texts/computing/security/SSL%20And%20TLS%20Essentials%20-%20Securing%20The%20Web%202000.pdf>

The Smashing Team. 2011. "Responsive Web Design – What It Is And How To Use It." *The Smashing Editorial*. Published January 12, 2011. <https://www.smashingmagazine.com/2011/01/guidelines-for-responsive-web-design/>

Thomas, Bostrom Jorgensen. 2014. "Native vs HTML5 security: is there a third way for app development?" *TechRadar.pro. News*. Published November 13, 2014. <http://www.techradar.com/news/software/applications/native-vs-html5-security-is-there-a-third-way-for-app-development-1272900>

VisionMobile Report. 2013. "Developer Economics 2013." *Developer Tools: The foundation of the App Economy*. Published January 2013. Page 25. <https://www.scribd.com/document/121779311/visionmobile-report>

W3C. 1999. "JSON - Introduction." *w3schools.com*. Published 1999-2017.

https://www.w3schools.com/js/js_json_intro.asp

Windows Dev Center. 2017. "Get started with Windows apps." *Microsoft, Build UWP apps that work on all Windows 10 devices*. Updated 2017. <https://developer.microsoft.com/en-us/windows/apps/getstarted>