Financial Management System

Final Project Proposal

By

J. Michael Kingston
Steven Miller
Isaac Kingston
Joseph C. Kingston

University of Utah, College of Engineering

May 9, 2009

TABLE OF CONTENTS

INTRODUCTION

The purpose of this project is to create a system that allows a user to extract their bank information, and compare this data with physical documents. The physical documents range from checks, receipts and other transactional documents. In this project, the bank information will be extracted through the internet, and the physical documents will be scanned using a check scanner. With these objectives in mind, the project design has been split into two main components: A Bank Information Extraction (BIE) component, and a Physical Document Extraction (PDE) component. These two components share a database that provides the capabilities necessary to share and compare data within this system. In the design process, the BIE and PDE components have been further broken into sub-components. Fig. 1 shows the BIE and PDE components, the database, and all of the sub-components that will exist within this project.

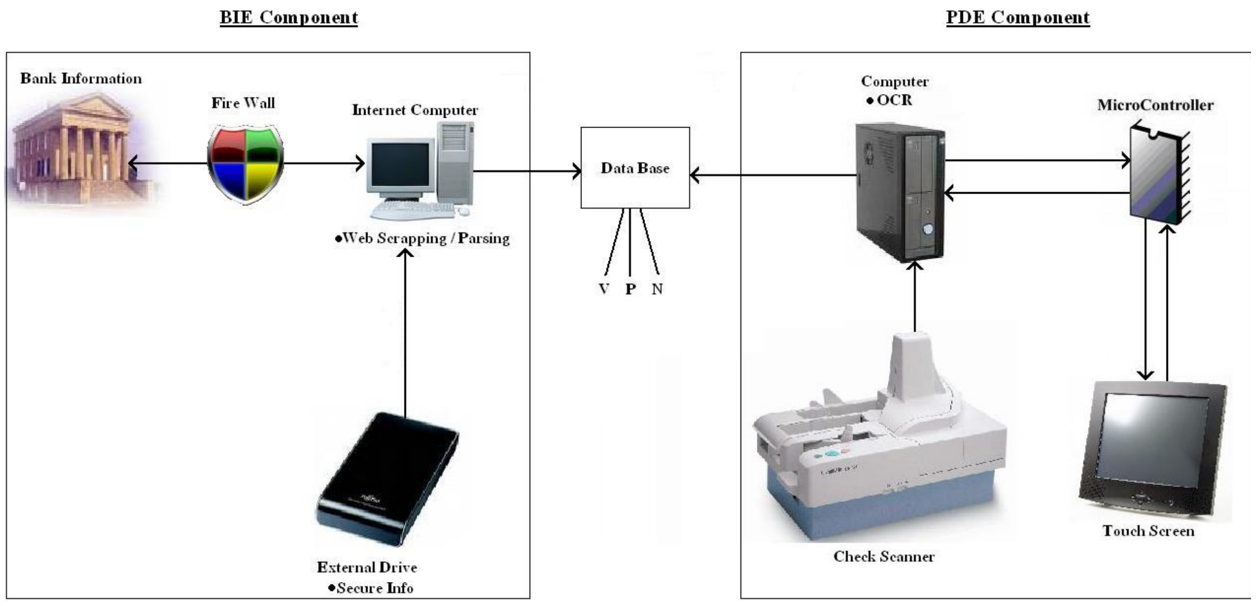


Figure 1. Diagram of project components.

As shown in Fig. 1 there are a lot of components that exist within this project. Because of this, the focus of this project will be more business oriented as opposed to personal use and benefit. In other words, this project will be constructed to work with a higher volume of data than what a single person or small group could provide.

BIE COMPONENT

The BIE component will provide a way for a user to insert their bank account information. This component will grab the user's transactional data from the bank's website. This component consists of three sub-components which will be managed by the administrator. These sub-components are: a firewall, an internet computer, and a secure external hard drive. A commercial firewall will be used strictly for providing a secure method of extracting bank information. This will limit malicious users and programs from accessing this data. As shown in Fig. 1, this will be the sub-component that works with the internet computer, and the banks through their website. The internet computer is going to be responsible for extracting the bank information. It will have a server running which will control the web-programs responsible for extracting this bank data. This computer will not store any secure bank information that could potentially be stolen through the web. Instead, all of this data will be stored on an external hard drive. This external drive will be responsible for safely storing all of the bank account information given by the user. Everything on this drive will be encrypted, and only decrypted when needed by the server program. All information on this drive will be backed up in case it is lost or stolen.

PDE Component

The PDE component is responsible for providing a user interface which allows a user to scan physical documents, extract the data that exists on them, and confirm or deny that this data should be sent to the database. The PDE component consists of four sub-components: a computer with Optical Character Recognition (OCR) software, a check scanner, a microcontroller, and a touch-screen. The OCR computer will use commercial OCR software to extract the data from the scanned documents. As shown in Fig. 1, the check scanner will be connected to the OCR computer. This scanner will send the images of each check-sized document to the OCR computer. The OCR computer will attempt to translate the data, and acknowledge translation errors identified by the OCR software. The OCR computer will receive commands from the user through a touch screen controlled by a microcontroller. Thus, this computer is responsible for providing everything the microcontroller needs in order for it to correctly prompt and respond to the user. This computer will also be responsible for sending the data to the database when requested by the user. The microcontroller will be responsible for providing a user interface for the user via a touch screen. It will process all of the input received through the touch screen, then let the user know the status of the check scanner and OCR software. It will prompt the user to confirm the data is correct, and provide a way for the user to fix incorrect data otherwise.

BIE Sub-Component Details

The functionality and implementation of the three BIE sub-components are described in detail below.

Firewall

The BIE Firewall will be installed on the internet computer. The commercial firewall that will be used for this project will be a MacAfee firewall. This firewall will protect the internet computer by denying any unauthorized access to external users. This firewall has complete inbound and outbound security. It is also programmed to automatically trust known applications, block spyware, Trojans, and key loggers (McAfee, 2009).

Internet Computer

The internet computer can be thought of as the central hub for the BIE portion of the project. Every single part ties in with this sub-component. First, all traffic from the world, either in or out, will run through the internet computer. This will happen through a program we write that will be housed on the internet computer. This program will be referred to as the BIE Communications program. Next, the internet computer will hold the program responsible for encrypting and decrypting the information on the secure flash drive. The purpose for encrypting the data is so that private information won't be compromised by spoofers and hackers, or in the rare case that the drive gets lost. The internet computer will also contain firewalls between the internet computer and the internet along with a firewall between the database and the internet computer. Lastly, the internet computer will be responsible for any data traffic into our system from the internet.

One important design issue that has been decided is to not store any data directly related to financial or sensitive information on the internet computer. The extent of what is stored on the computer will be what it takes to make the programs work and interface with one another and the internet. This was decided as a security measure against malicious users that might probe the internet computer.

BIE Communication Program

This program is responsible for retrieving the bank information to compare with scanned documents. The software to be written for this will be a server/client web-based program. This program will navigate and collect information from the internet. There are two high-level ideas that will be used to grab the data from the internet. These ideas are scraping data manually and grabbing the data from a nicely packed API. Scraping is a widely known technique for pulling information from a webpage. Scraping will appear to others viewing it as just another human accessing their web site, but in reality it is a computer program. Different behaviors scraping simulates are: regular expression matching, embedded semantic recognition, html parsing, browser control, http programming, and human copy-and-paste (Shrenk, 2007). Some sites want people to scrape their websites while others despise scrapers. A site that might welcome this technique would be an online bookstore such as Amazon that wants to get its information out to as many people as possible. A site that would discourage this would be one that had secure information or a business that was providing a service; they wouldn't want someone to steal their information and pose it as their own. The sites we will be pulling data from will be bank websites which hold extremely secure information. Scraping is the hardest of the two methods mentioned above and poses different problems. First, if the banks made even the slightest change to their website, which they do all the time, then the scraping software would have to be modified. Next, trust can be a big issue. Banks may be uncomfortable with unknown users crawling their site. The reason scraping would be used is if the API being provided didn't provide all of the needed information.

The second and more preferable method is grabbing information from a provided API. An API is a nicely packaged interface that the banks would provide for our system to grab and parse bank information. There already exists products that require web-based bank extraction. To satisfy these requirements, banks provide them with an API containing their requested data. Some of these products are mint.com, Peachtree, and Quickbooks. In particular, Quickbooks uses a protocol for their API called direct-connect. The financial management project would use a similar protocol. One main difference between the financial management project and some existing products is the financial management project is designed at a higher level of automation. The plan is to create a tool that operates at the click of a button. The project discourages manual entering at a high level. The banks will provide the financial management system with an API.

The web-architecture portion of the program will be deployed using Apache's free Tomcat server. Some of the tools to write the actual program are outlined below:

<u>Java Servlets</u> A Java servlet is a programming language object that dynamically handles requests and responses on a server. It uses the Java programming language as its platform. The servlet lifecycle consists of initializing the servlet object, servicing requests to and from a server, and closing the object once it's finished. These servlets will constitute the heart and main functionality of the BIE Communications Program.

<u>Java JSP:</u> A JSP is very similar to a Java servlet. At compile time it compiles into a Java servlet, but at the programming stage it has an interface that is preferable for different tasks. It uses a mixture of html, pseudo Java, and embedded packages to make programming for different aspects easier. Usually for graphical displays and style of a webpage, it's easier to accomplish this through a JSP, while logic and database traffic is better suited for a Java servlet. The project will use JSPs to make the BIE Communications Program look nice.

<u>CSS:</u> CSS stands for Cascading Style Sheets. It is a language that describes the presentation of a document. It meshes directly with the html programming language. One of the nicest features of CSS is providing a central location for all style aspects of a web-page. CSS is very flexible using a priority system based on object matching. This along with the JSPs should give the project a crisp look.

<u>HTML:</u> HTML stands for Hyper Text Markup Language. It is the true programming language of the World Wide Web. It is the protocol for communicating over the internet. It provides the basics for web programming, but lacks many features that Java servlets and other tools provide such as dynamic interaction. Html will be embedded in nearly every aspect of this program.

<u>XML:</u> XML is short for Extensible Markup Language. It is a general protocol for creating user defined markup languages. It is primarily responsible for sharing structured information across various platforms. XML ties in very nice with Tomcat and is useful for installing security measures. This is how the project will incorporate internet encryption. This is also how the

project will include user-defined URLs, Welcome page options, and any filtering that needs to be done.

Along with the basic functionality of the BIE Communications program, some nice features will be implemented such as emailing users when accounts are overdrawn and various other financial management tools.

External Hard Drive

As mentioned above, the design decision was made to not store any sensitive data on the internet computer. This was a security precaution put in place, but this also creates other issues. The biggest issue that has risen is the need to access secure information such as passwords and bank account numbers. The solution for this was incorporating an external drive to hold any secure information needed by the internet computer. When the program needs to access the data on this drive, the user will be prompted for a password. If a valid password is provided then the program will take requested data and send it to the banks in the common encryption standard Secure Socket Layer (SSL). When the program is finished using the drive, the drive will be physically disconnected so that there won't be any accessible trace to the information.

The type of drive being used is the Iron Key brand. Iron Key claims to be the most secure flash drive in the world. Encased within the flash drive is a chamber of acid. To access the drive, one needs to supply the correct password and usernames. The drive allows so many attempts to access it. When this limit is met, the acid chamber breaks and bathes the drive in acid. This would eliminate any worries if the drive got lost or if someone tried to access it from the internet.

External Flash Drive Encryption Program

This portion will be referred to as EFDEP. This program will be housed on the internet computer. The two main reasons for this program are interest in encryption and extra security precautions. This would stall any hackers. Also, if someone were to access the drive without proper permission, then they wouldn't be able to read the data. The encryption being used will be Rivest-Shamir-Adleman (RSA). One of the nice features of RSA is it's a public key protocol. This means that there are two separate keys for encryption and decryption. If someone got access to the public key then they wouldn't be able to decrypt the data without the private key. Another portion of the encryption method that will be used for this portion of the project is random assignments of keys. This is one of the most secure encryption schemes ever devised because there is no mathematical way to break it. The only problem is the keys have to be transferred from one location to another. Since our system is self-contained then this eliminates the problem. This would be implemented using a random 40 digit key. The mapping to the keys would be accessible to the internet computer. Periodically the keys would be changed. Whenever the internet computer needed data from the drive it would request authentication. Once the drive is accessed appropriately then the internet computer would process the data appropriately.

DATABASE

Ingres provides an open source version of their database management software. This version should be sufficient for the project purposes. However, if additional functionality is needed, Fidelity Funding has offered use of their Ingres software for the time of this project.

DATABASE USER INTERFACE

In order to put all the database information at the user's fingertips and display this information in a meaningful way, a user interface must be created for the database. This program will provide the user with a table view of all transactions, graphs displaying expenditures, and a printer friendly report of one or more accounts. This program is strictly for viewing purposes only. The user will not be able to make changes to the database. The user interface's purpose is to make information in the database easily accessible and printable for the user.

One may ask, if the user sees an error in the database, shouldn't he/she be able to fix that error? This is actually a tough question and leads to one of the main differences between QuickBooks and Peachtree. QuickBooks allows the user to edit transactions after they have been entered into the database while Peachtree does not. Peachtree's approach eliminates the possibility of employees falsifying records (Pratt, 2009). For this project, the only way to make changes to the database is by editing the database manually through Ingres database software.

The user interface will be programmed using web architecture and Java technologies as mentioned previously. This is simply because the authors are most familiar with creating user interfaces through web design. From the main screen the user will be given information on all transactions in the database. They will be able to narrow down the information by selecting from a list of businesses, and the date range they are interested in.

Once a business is selected, the transaction data will load into a table much like that in Table 2 below. There will also be a table summary of the different accounts for that business like that seen in Table 3. One graph will display a year to date summary of how this business is doing like that seen in Fig. 3. A pie graph like that in Fig. 2 (Mint.com) will show a summary of where all money was spent. A separate pie graph will show all income categorized by the type of income. The categories in these pie graphs will be clickable to view where the transactions within that category originated. For example, the user can click on the "Groceries" category and view what percentage of business groceries came from Wal-Mart vs. ShopKo, etc.

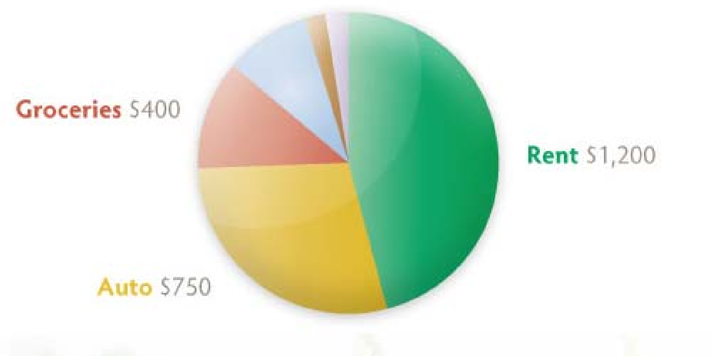
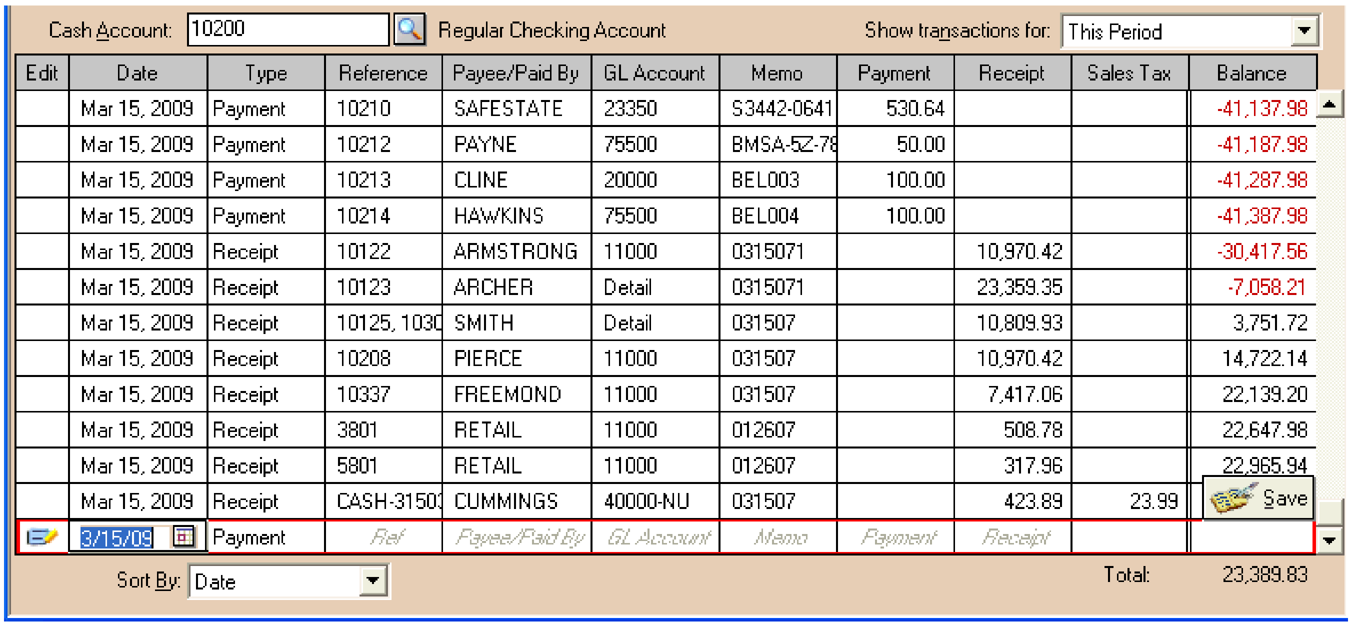


Figure 2: Clickable Expense & Income Pie Graphs

.

Table 2: Transaction Table

Below is a sample transaction table taken from Peachtree accounting software. This table will be sortable by date, payee, or account. The user will be able to click on a transaction to view an image of the scanned document. Minor changes will be made to the table below. The table will not be editable and transactions cannot be added via this interface. The balance and Sales Tax columns are not necessary for this project's purposes. The Memo column will be changed to "Description" (Peachtree Free Trial).



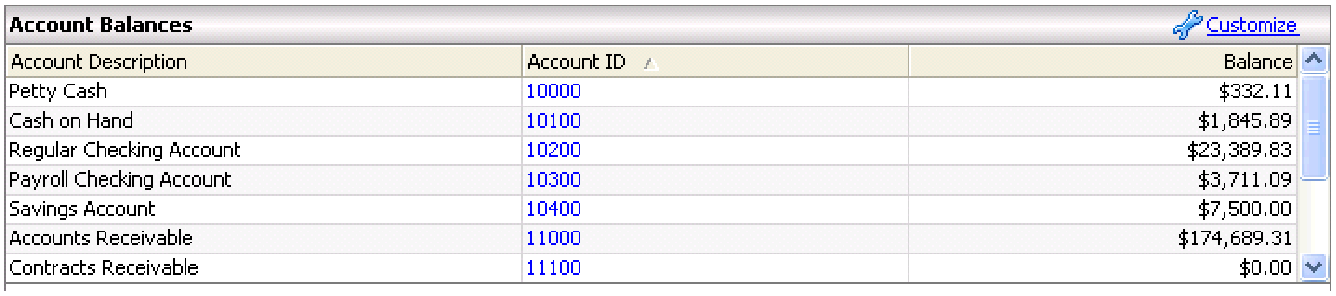
Cash Account: 10200 Regular Checking Account   Show transactions for: This Period

Edit	Date	Type	Reference	Payee/Paid By	GL Account	Memo	Payment	Receipt	Sales Tax	Balance
	Mar 15, 2009	Payment	10210	SAFESTATE	23350	S3442-0641	530.64			-41,137.98
	Mar 15, 2009	Payment	10212	PAYNE	75500	BMSA-5Z-78	50.00			-41,187.98
	Mar 15, 2009	Payment	10213	CLINE	20000	BEL003	100.00			-41,287.98
	Mar 15, 2009	Payment	10214	HAWKINS	75500	BEL004	100.00			-41,387.98
	Mar 15, 2009	Receipt	10122	ARMSTRONG	11000	0315071		10,970.42		-30,417.56
	Mar 15, 2009	Receipt	10123	ARCHER	Detail	0315071		23,359.35		-7,058.21
	Mar 15, 2009	Receipt	10125, 1030	SMITH	Detail	031507		10,809.93		3,751.72
	Mar 15, 2009	Receipt	10208	PIERCE	11000	031507		10,970.42		14,722.14
	Mar 15, 2009	Receipt	10337	FREEMOND	11000	031507		7,417.06		22,139.20
	Mar 15, 2009	Receipt	3801	RETAIL	11000	012607		508.78		22,647.98
	Mar 15, 2009	Receipt	5801	RETAIL	11000	012607		317.96		22,965.94
	Mar 15, 2009	Receipt	CASH-3150	CUMMINGS	40000-NU	031507		423.89	23.99	
	3/15/09	Payment	Ref	Payee/Paid By	GL Account	Memo	Payment	Receipt		

Sort By: Date   Total: 23,389.83

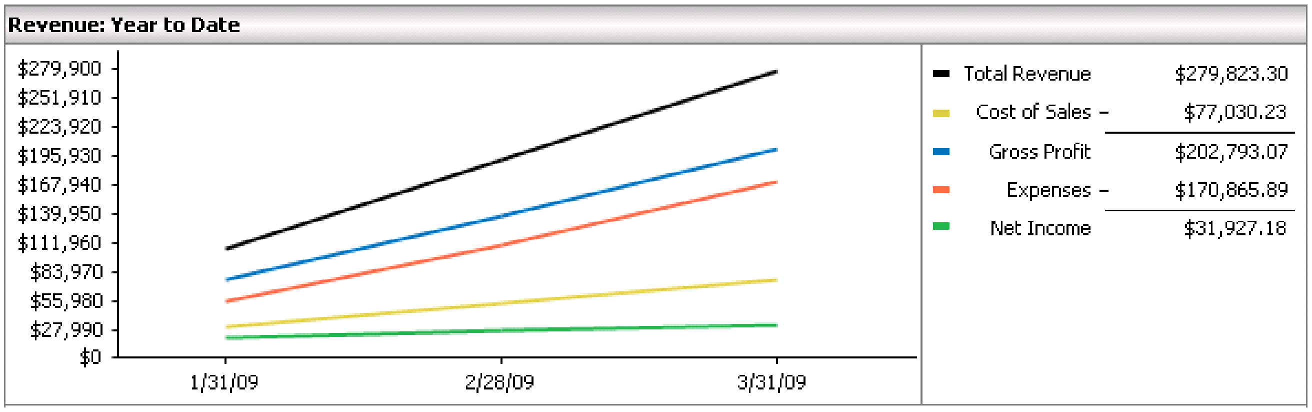
Table 3: Account Balances

Within a business there may be several different accounts. Whether they are Cash vs. Check accounts or simply separate bank accounts, this table displays the balances in each account. The Account ID will be the primary key of the table (Peachtree Free Trial).



Account Balances Customize

Account Description	Account ID	Balance
Petty Cash	10000	$332.11
Cash on Hand	10100	$1,845.89
Regular Checking Account	10200	$23,389.83
Payroll Checking Account	10300	$3,711.09
Savings Account	10400	$7,500.00
Accounts Receivable	11000	$174,689.31
Contracts Receivable	11100	$0.00



■	Total Revenue	$279,823.30
■	Cost of Sales –	$77,030.23
■	Gross Profit	$202,793.07
■	Expenses –	$170,865.89
■	Net Income	$31,927.18

Figure 3: Revenue Year to Date

This graph is an easy way to tell how the business is doing overall. The legend to the right lists the exact amount in each category. A bar graph may be more desirable for this figure (Peachtree Free Trial).

Database Security (VPN)

To maintain maximum security of sensitive bank information, read and write access to the database must be delegated carefully. A Virtual Private Network (VPN) will be set up as the only network that will have read access to the database. This network will be disconnected from the internet to prevent hackers from getting information from the database. Access to the database will be shielded from internet computers by a firewall that will allow data to be written to the database, but not read from the database. With this safeguard the worst thing hackers could do is write to the database. They would not be able to read any sensitive information because all connections to internet computers are one way connections. To ensure that no unauthorized users write to the database from an internet computer the user will have to log in before she can write to the database.

PDE Sub-Component Details

The functionality and implementation of the four PDE sub-components are described in detail below.

OCR Computer

The OCR computer is going to be connected to the database, the check scanner, and the microcontroller. From a user's point of view, this computer will receive images from the scanner, send and receive information to the microcontroller, and send information to the database. From an implementation point of view, The OCR computer has two distinct objectives. The first and hardest objective is to translate scanned text (both written and printed) into machine-editable text. The second objective is to provide interfaces for the microcontroller and the database.

Translating printed text into machine-editable text has been a difficult task for software engineers, yet in recent years is now considered a largely solved problem. To date, typical accuracy rates have been known to exceed 99%, however certain applications which demand higher accuracy require that people review the results for errors. One important note to make is that accuracy rates are measured in different ways. The two most common are character-by-character error rates, and full word error rates (Suen, 1998). For this project, the most optimal of OCR software would be one that could translate printed and handwritten text with as high of an accuracy rating as possible. For this system, its fine to require the user to review possible mistakes as this will be accomplished through the interface with the touch screen. Although it would be nice to have software that recognizes cursive, this is still an active area of research. Of the existing software that's on the market, cursive recognition rates are noticeably low. For the most complex recognition problems, intelligent character recognition (ICR) systems are generally used, which is another growing area of research. Within this technology, artificial neural networks can be made indifferent to both affine and non-linear transformations (Le-Net, 2009). This is why the design decision was made to use commercial OCR software instead of creating it ourselves.

Within the scope of this project, the main cost factors have been broken down. The four main factors are whether or not the software:

- recognizes handwritten text.
- supports dynamic form layout.
- allows a program to work with it as opposed to an actual user
- supports high-volume based applications.

With each of these factors, there is an increase of a factor of ten in the cost. Table 4 consists of all the OCR software that has been considered for this project as well as the actual software that will be used for the project (the one that's bolded in the table).

Table 4: Options for OCR Software[1]

Vendor	**Type**	**Cost**	**Interpret Handwritten Text?**	**Dynamic Form Layout?**	**Program-Friendly?**
Abbyy	FineReader 9 Corporate Edition	$554	No	No	No
Abbyy	**FlexiCapture 8.0 Professional (Full Version) – Unlimited**	**$14750**	**Yes**	**Yes**	**Yes**
Abbyy	FlexiCapture 8.0 Professional (Full Version) – 5,000 Pages per Month	$5900	Yes	Yes	Yes
Abbyy	FlexiCapture 8.0 Professional (Fixed Form Only) – Unlimited	$7200	Yes	No	Yes
Abbyy	FlexiCapture 8.0 Professional (Fixed Form Only) – 5,000 Pages per Month	$2950	Yes	No	Yes

According to the specs of this software, FlexiCapture 8.0 Professional (full version) unlimited can satisfy all of the features needed for this project including translating handwritten text. However, the next step will be to create full functioning tests to confirm this.

The other objective of the OCR computer will be to effectively communicate with the database and the microcontroller. This design concept is simple. This computer will need to be able to retrieve command signals from the microcontroller. The controller may ask for document images, important OCR software information, or scanner status. The OCR computer will send information to the database once it receives confirmation from the user.

CHECK SCANNER

The check scanner that will be used in this project is a CR-180 check scanner from Canon™. The only sub-component that this scanner will be connected to is the OCR computer. This scanner will be responsible for providing a way for the user to scan their physical documents, and to send the images to the OCR computer. According to the datasheet for this scanner, it can scan 180 checks per minute at 200 dpi in black and white or gray scale. The largest document it can scan is 9.1” X 4.6”. The smallest document this scanner can scan is 2.4” X 4.6”. It can scan paper with all standard weights including regular check paper, NCR, cardstock, and receipt paper as long as the rollers are adjusted ahead of time. The scanner can also scan papers of multiple sizes at the same time. The scanner also has a built-in Magnetic Ink Character Recognition

[1] This table information came from tech support person Cheryl Salemy who works with ScanStore.com

(MICR) component that will simultaneously read the magnetic ink on each check. This scanner has all of the capabilities that are required for this project.

MICROCONTROLLER / TOUCH SCREEN

The goal of the microcontroller and touch screen is to provide a user interface to have some sort of control over the scanner. The touch screen will also display the status of the scanner to the user.

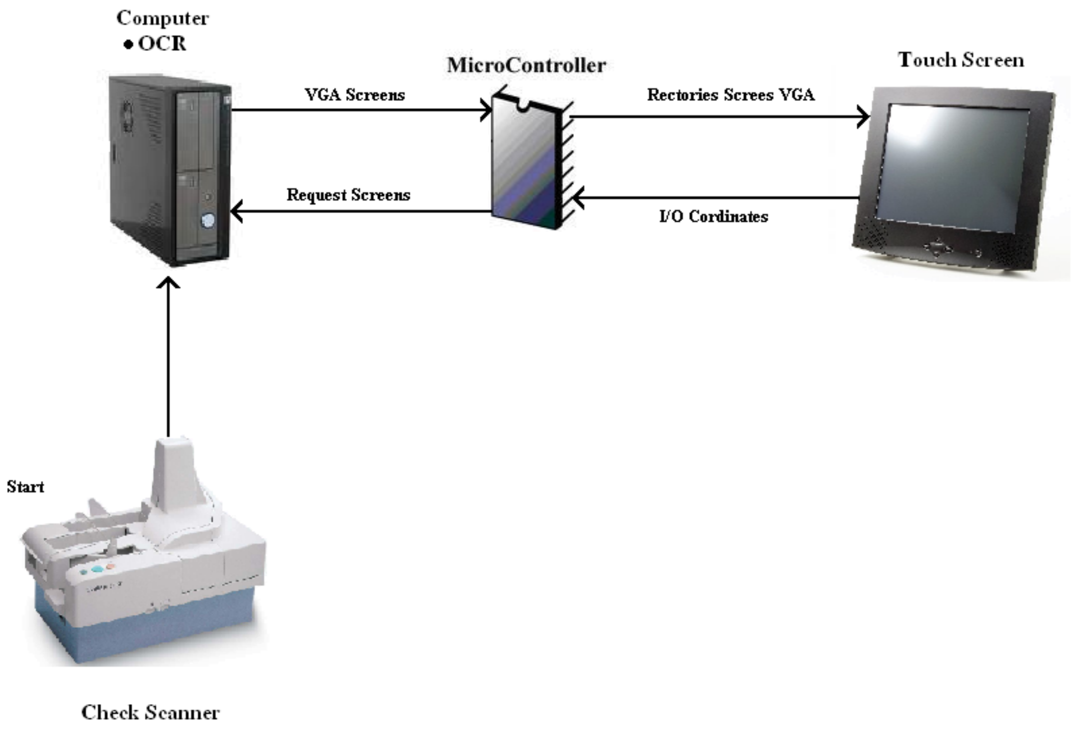


Figure 4. PDE Component

PDE USER INTERFACE

The scenario of the PDE component will run as follows: When the scanner is idle or off the microcontroller will remain idle, continually checking its input connection with the OCR computer for the status of the scanner. During this time, the touch screen will be turned off. When the scanner is being used by the user, the OCR computer will alert the microcontroller. The microcontroller will then activate the touch screen displaying that the scanner is currently running. As the documents are being scanned, the amount of documents being scanned will be counted and the amounts of each document will be added up into a batch total.

Once the scanning process is complete with no errors, the OCR computer will send the amount of documents scanned and a batch total to the microcontroller. After this, the OCR computer will wait for a confirmation from the microcontroller before it sends the scanned data to the database. Once the microcontroller receives the number of documents scanned and the batch total, the microcontroller will send a screen out to the touch screen. It will then open its I/O port and wait for input from the touch screen. The touch screen will then accept the screen from the microcontroller. The touch screen then displays that scanning has completed successfully and will display the number of documents scanned and the total amount of the batch. The touch screen will then ask for the confirmation of the scanned items from the user. Once the confirmation is prompted by the user, the touch screen will send the confirmation to the microcontroller. The microcontroller will then send the confirmation to the OCR computer. When the OCR computer receives the confirmation it will send its data to the database.

In the case that the OCR software is unable to interpret data on the scanned document(s), the following events occur. The OCR computer sends the image of the document and the error to the microcontroller. The microcontroller takes the document image and creates a screen with the document and a keyboard. Fig. 5 shows this screen. The touch screen uses this screen to display to the user that the OCR computer is unable to interpret the text of the document. The touch screen prompts the user to manually enter the information of the document on the keyboard that is displayed on the touch screen. Once the user enters in the correction, the data is sent from the microcontroller to the OCR computer while the scanning process continues.

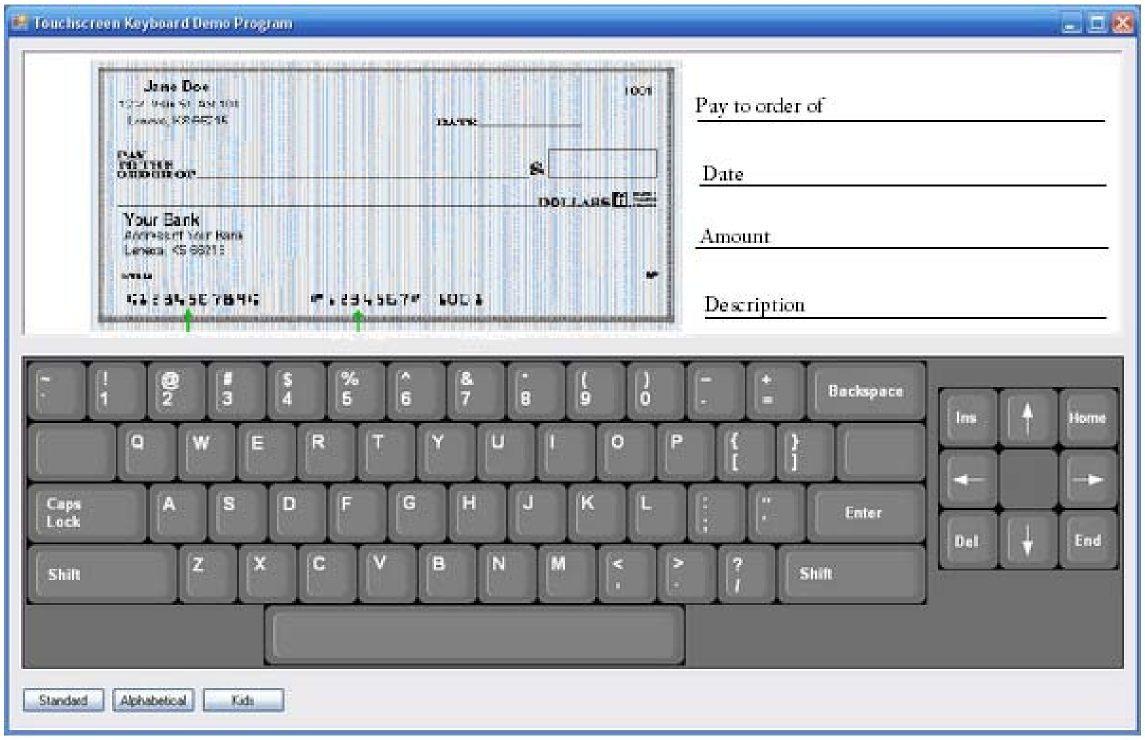


Figure 5. Document verification screen.

DISPLAY SCREENS

Screens that will be stored in the microcontroller are: the ready screen, the scanning documents screen, error screens, and the confirmation screen. The ready screen displays that the system is ready to begin. The scanning documents screen informs the user that documents are running through the scanner. There are two types of errors that the touch screen is responsible for displaying. The first error is the jamming error. The jamming error requires a screen that displays the location and condition of the jam. Once the jam is fixed, the process resumes automatically. The second error occurs when the OCR software is unable to translate the data on a document. A screen is displayed that shows a picture of the keyboard, and prompts the user to manually enter in the information. The final screen displayed is the confirmation screen. The confirmation screen states that scanning is complete and no errors have occurred. It also has a confirmation button for the user to verify the document batch total. Once the user confirms the batch, the ready screen comes up again.

Parts Needed

The parts needed for the touch screen interface are a touch screen and a microcontroller. The screens that are displayed on the touch screen are developed in the microcontroller. The microcontroller is re-programmable. The microcontroller used in this project is the ezLCD-004 which is shown Fig. 6.

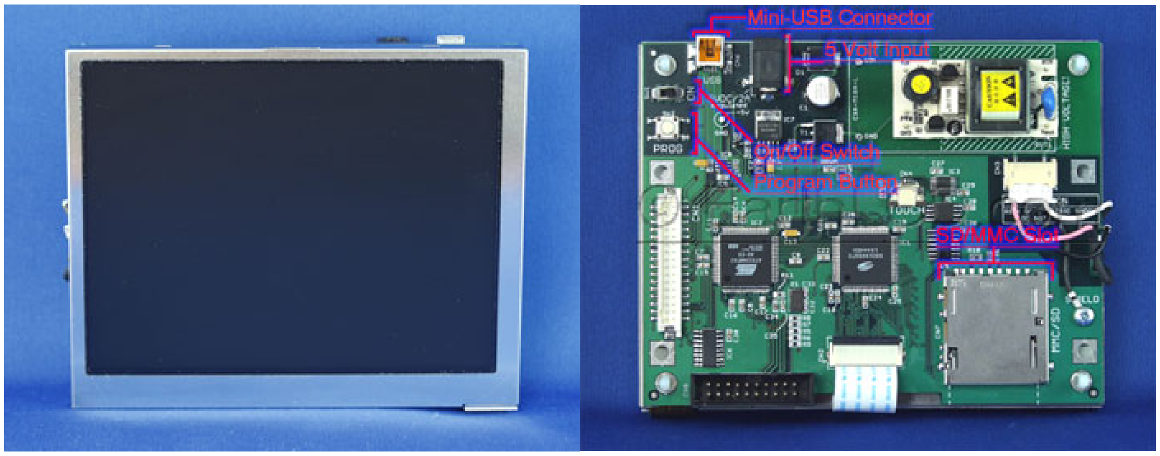


Figure 6. ezLCD-004 Touch Screen/Microcontroller Kit

Programming the microcontroller

The ezLCD-004 is manufactured by LCDEarth. The ezLCD-004 touch screen kit consists of a microcontroller and a touch screen. The features of the microcontroller are shown below:

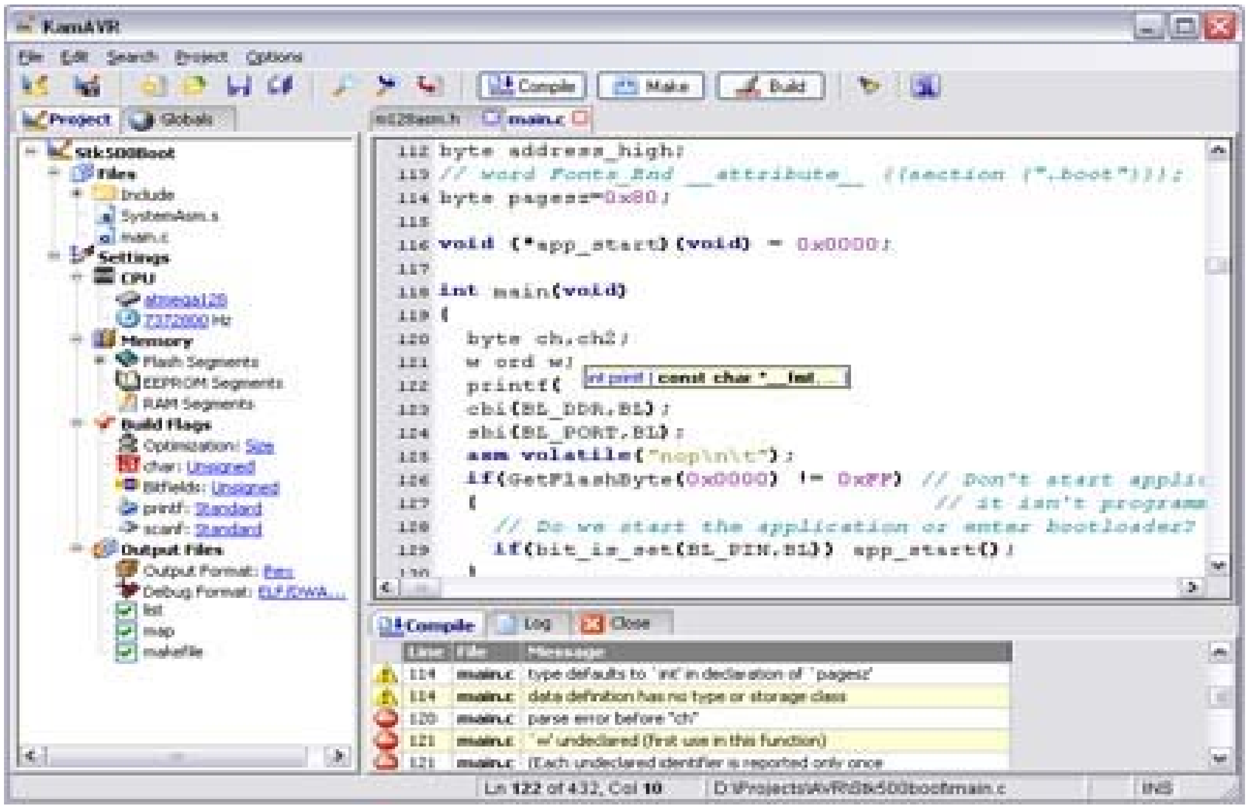
Microcontroller Features

- Full ezLCD Command Set
- 5.6" 16 bit Color TFT LCD, capable of 65,536 simultaneous colors
- Integrated Touch Screen *(resistive)*
- Dedicated LCD Controller
- Integrated SD/MMC (Secure Digital/Multimedia) Memory Slot
- Full Speed USB Interface (via Mini-USB connector)
- 128k Bytes Flash
- ARM Microcontroller

- Runs on 5 volts
- Display is transflective sunlight readable
- Programmable Display Module

This touch screen kit is ideal because of the support and development kit that's offered for this microcontroller and it's easy to learn and use.

LCDEarth provides a development kit with excellent support and documentation. The microcontroller is programmed and accessed through this development kit using the C language. The development kit also supports many different hardware interfaces such as USB and parallel/serial ports. This kit gives programmers the ability to control when the touch screen turns on, what is displayed on the screen, and what to send out to other devices and components. The development kit allows the programmer to simulate programs as they would appear on the touch screen. Once the program is compiled it can be downloaded through the development kit to the microcontroller. Fig. 6 shows a screen shot of the development kit in use (LCDEarth, 2009).



Stk500Boot
- Files
  - Include
  - SystemAsm.s
  - main.c
- Settings
  - CPU
    - atmega128
    - 7372800 Hz
  - Memory
    - Flash Segments
    - EEPROM Segments
    - RAM Segments
  - Build Flags
    - Optimization: Size
    - char: Unsigned
    - Bitfields: Unsigned
    - printf: Standard
    - scanf: Standard
  - Output Files
    - Output Format: Ihex
    - Debug Format: ELF/DWA...
    - list
    - map
    - makefile

```
112 byte address_high;
113 // word Fonts_End __attribute__ ((section (".boot")));
114 byte pagesz=0x80;
115
116 void (*app_start)(void) = 0x0000;
117
118 int main(void)
119 {
120   byte ch,ch2;
121   w ord w;
122   printf(  int printf(const char *__fmt, ...)
123   cbi(BL_DDR,BL);
124   sbi(BL_PORT,BL);
125   asm volatile("nop\n\t");
126   if(GetFlashByte(0x0000) != 0xFF) // Don't start applic
127   {                                 // it isn't programm
128     // Do we start the application or enter bootloader?
129     if(bit_is_set(BL_PIN,BL)) app_start();
130   }
```

```
Line  File    Message
114   main.c  type defaults to `int' in declaration of `pagesz'
114   main.c  data definition has no type or storage class
120   main.c  parse error before "ch"
121   main.c  `w' undeclared (first use in this function)
121   main.c  (Each undeclared identifier is reported only once
```

```
Ln 122 of 432, Col 10    D:\Projects\AVR\Stk500boot\main.c    INS
```

Figure 6. Development kit for the microcontroller

The screens will be designed on the development kit provided for the microcontroller. Each screen is designed with properties that explain the intent of the screen. The screen accepts I/O coordinates from the user and returns data back to the microcontroller. The data that will be sent back to the microcontroller can range from what buttons a being bushed to the actual I/O coordinates that are being pushed from the user. The development kit allows the programmers to create screens that include buttons, keyboards, and pictures for displaying. In order for the screens to work with the program, the screens will be designed and ready to use before the touch screen is running. Once the screens are designed, they will be downloaded into the microcontroller from the development kit. They will then be available to pull and display through the programming of the microcontroller.

INTERFACE SPECIFICATIONS

As mentioned before, there are a lot of components that exist within this project. There are two main integration specifications to take care of, and those are: software integration and hardware integration. Because all of the hardware used in this project is commercial, and none of it will be designed at the circuit level, the hardware integration will be straight forward. Fig. 7 below shows all the hardware interface specifications that will exist within this project.

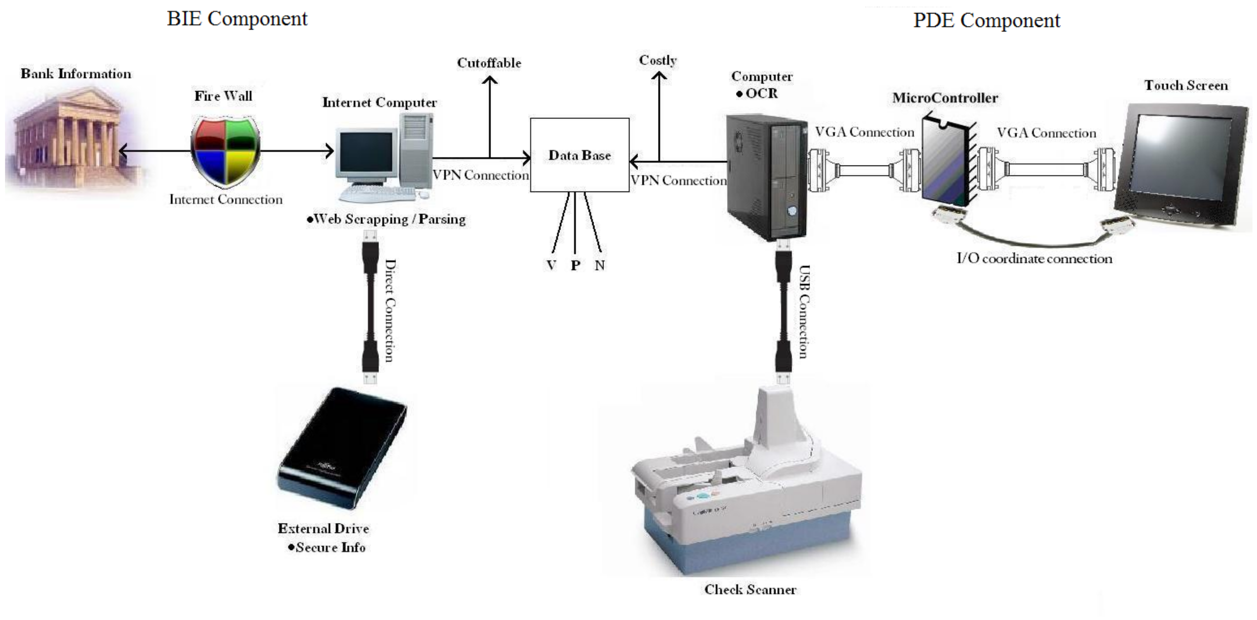


Figure 7. Diagram of the project layout with the interfaces specified.

Once each component is plugged in, all of the hardware drivers will be automatically installed if they aren't already. Each component will have all the power cords and features needed to operate them. The software integration, however, will take a lot more work.

INTEGRATION WITHIN THE BIE COMPONENT

The internet computer is where all of the main software integration will occur for the BIE component. It will be responsible for integrating with three portions of the project which are: the internet, the database, and the external hard drive. On this computer, there will be a program controlled by the server that will integrate with bank websites in order to extract the needed information. As mentioned before, this will happen using an API provided by the bank. The program on the internet computer will need to interact with the external drive to retrieve bank account information. Once the internet computer has all of the needed information from the banks, the external drive will be disconnected, and the computer will send this bank information to the database using a one-way VPN connection (as seen in Fig. 7). This one-way connection will be enforced through software. Also mentioned before, a program on the database computer will identify the internet computer, allow it to store data, and block any retrieve requests.

INTEGRATION WITHIN THE PDE COMPONENT

The OCR computer is going to take care of the majority of the software integration for the PDE component. This computer will begin receiving roughly 180 images per minute from the check scanner once the user starts scanning. Software on this computer will need to automatically identify that the scanner is in use so that it can prompt the microcontroller to output the correct interface to the user. Also, as the images are coming in, the integration software on this computer needs to start up the OCR software, and begin translating the text that exists on the documents. Once the OCR software is finished, this computer will identify if there were any translation errors, and forward this information to the microcontroller. As long as the microcontroller receives the data it needs from the OCR computer, it can easily do its job of providing a usable interface for the user through the touch screen. This microcontroller was made to work specifically with the touch screen that will be used. For this reason, integration between the microcontroller and touch screen will be straight forward. Once the user is finished and confirms that the data should be sent to the database, the integration software on the OCR computer will identify this, and send over the data through a one-way VPN connection. Similar to the one-way connection for the internet computer, this will be controlled through software that exists on the database computer.

RISKS

There exist three main risks in this project: bank information extraction, security, and integration. Integration is more of a challenge than a risk. The challenge is getting all components in the project to work together seamlessly. Security is paramount to this project because it handles personal financial information. If security of the financial information is compromised this project will become a liability to its users rather than a service. Bank information extraction is only a risk because all banks may not provide an API for us to work with. If we must resort to scraping bank information several problems arise. These risks are currently being addressed and will be resolved before fall 2009.

The main problem inherent in scraping a bank website is change. Websites are always changing, and when they do, the scraper will break. It isn’t difficult to fix a scraper once it breaks, but doing so takes time. It would be a full time job maintaining scrapers for all the different banks. The first bank used for this project provides an API. However, it is unknown whether other banks will. Also, for the banks that provide an API, the API may not give all of the information needed. If this is the case, scraping may inevitable.

Security is a risk for any financial management system. Failures in security may only be apparent after a clever hacker steals user data. By thoroughly researching current security trends, the security of this financial system will continue to improve.

To meet the challenge of integration the authors will resolve integration issues as early on as possible. Rather than saving all integration for the end of the semester, the authors will integrate each piece into the project as soon as that piece is complete. With careful planning, when integration issues arise there will be time to deal with them. More details of the integration plan can be found below in the project schedule.

Many previous risks of this project have been resolved such as funding for the OCR software, and purchase of the Canon check scanner. Below is a bill of materials describing how each component of the project will be provided. The total cost of this project will be $350. This does not include the components already obtained and those provided by Fidelity Funding.

Bill of Materials

Table 5: Consists of all the materials needed for this project.

Item	**Price**	**Provided By / Purchased From**
Internet Computer	**$0.00**	**Steven Miller**
Database Computer	**$0.00**	**Joseph Kingston**
OCR Computer	**$0.00**	**Michael Kingston**
Ingres Database Software	**$0.00**	**Ingres (open source)**
OCR Software	**$0.00**	**Fidelity Funding**
Microcontroller/LCD Kit	**$350.00**	**LCD Earth**
Canon CR-180 Check Scanner	**$0.00**	**Fidelity Funding**

Schedule Flow/Project Milestones

The list below shows the current schedule that we will follow for this project:

- First Day of Fall Semester (Monday, August 24, 2009):
    - Isaac will have obtained the touch screen and microcontroller. The touch screen will be tested to make sure it works when given VGA input. The microcontroller will be programmed with a simple program just to make sure it works individually.

  - Michael will have the OCR software and check scanner obtained from Fidelity Funding. He will have tested a batch of 25 checks to see how the OCR software performs in interpreting the handwriting. He will have the OCR software and check scanner working together without the touch screen and microcontroller.
  - Steven will have a program successfully working with the Bank's API, and one bank account will be successfully downloaded into the database. Steven will begin enforcing the necessary security precautions and encryption.
  - Joseph will have one account stored in the database to write the Database user interface. He will have the basic shell of the user interface completed. He will have the pie graphs for expenses and income displaying one account's information. He will begin working on the Account Balances table.
- Monday, September 14, 2009:
  - Joseph will have the Transaction table complete and ready to demo. He will be nearly finished with Account Balances table.
  - Isaac and Michael will have the microcontroller and touch screen working together.
  - Steven will have the security constraints in place for the with the BIE component.
  - Michael will have a program running on the OCR computer that works successfully with the OCR software, and is able to communicate with the microcontroller and database appropriately.
- Monday, October 5, 2009:
  - Joseph will have the "Revenue Year to Date" graph working. He will start creating printer friendly reports.
  - Steven will have the encryption program written and functional. He will begin thoroughly testing this program.
- Monday, October 26, 2009:
  - Full testing and heavy integration will begin for all components of the project.
- Wednesday, November 16, 2009:
  - Everything will be completed. The Third draft of the project paper will be completed.

Fig. 8 below shows a Gantt chart of the entire schedule for the Financial Management System project.

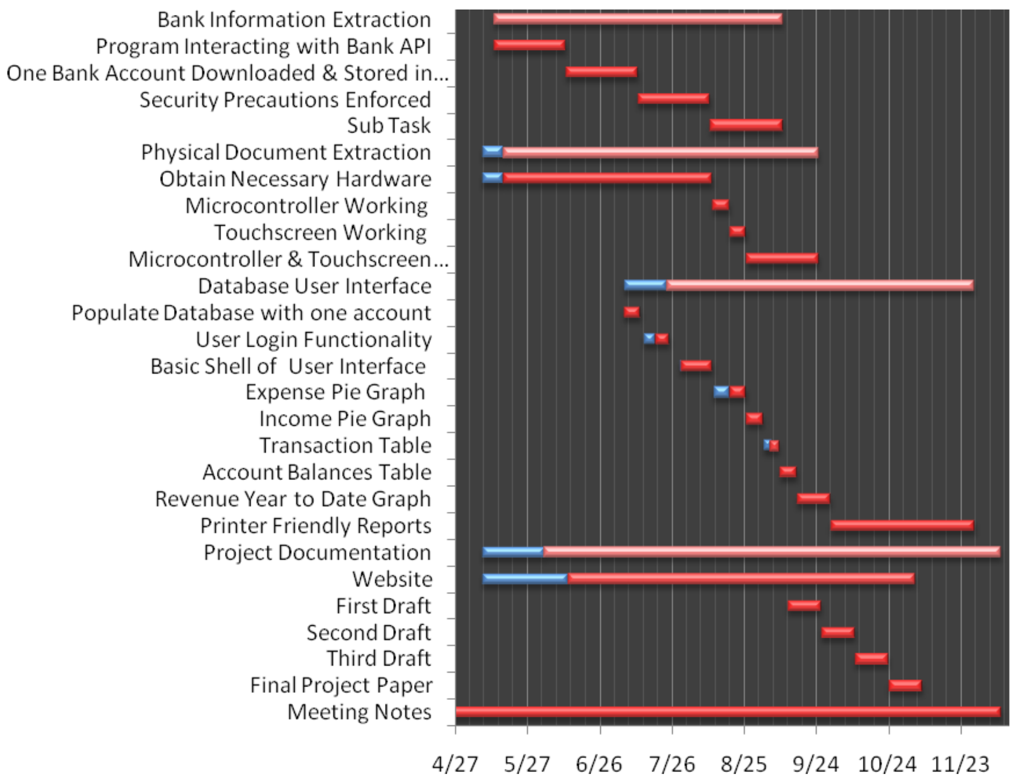


Figure 8. Gantt chart of the schedule for this project. Blue indicates that the task is that much complete, while red indicates that the task still needs that much more work.

BIBLIOGRAPHY

Canon. (n.d.). *imageFORMULA CR-180 Check Transport*. Retrieved April 16, 2009, from www.usa.canon.com/opd/controller?act=OPDModelDetailAct&fcategoryid=2538&modelid=9468

Free Trial. (n.d.). Retrieved May 1, 2009, from Peachtree: www.peachtree.com

LCDEarth. (n.d.). *ezLCD*. Retrieved May 2nd, 2009, from www.ezlcd.com

Le-Net. (n.d.). *Conventional Neural Networks*. Retrieved April 30, 2009, from LeNet-5 Demos: http://yann.lecun.com/exdb/lenet/

McAfee. (n.d.). *Firewall*. Retrieved April 30, 2009, from security wordbook: http://www.mcafee.com/us/security_wordbook/firewall.html

Mint.com. (n.d.). Retrieved April 28th, 2009, from www.mint.com

Pratt, N. (2009, April 26). Database Management. (F. M. Team, Interviewer)

Salemy, C. (n.d.). Retrieved April 6, 2009, from ScanStore: www.scanstore.com

Shrenk, M. (2007). *Webbots, Spiders, and Screen Scrapers.* No Starch Press.

Suen, C. Y. (1998). Future Challenges in Handwriting and Computer Applications. *3rd International Symposium on Handwriting and Computer Applications* , 9.