Prior Learning Assessment Application

PLA Type	**Portfolio Review**
Student	[redacted]
Student UC ID	[redacted]
Student Email	[redacted]
Student Phone	[redacted]
Submission Date	[redacted]
Target Course	[redacted]
Degree Program	[redacted]
College	College of Education, Criminal Justice, and Human Services (CECH), School of Information Technology
Referring Faculty	Professor [redacted]

Contents

Introduction

My goal with this PLA portfolio application is to make a case that I have attained the Learning Outcomes of the course [redacted] through real-world on-the-job experience outside the classroom.

This is in pursuit of the University's "Prior Learning Assessment" option that allows students to:

- earn college credit.
- meet the CECH SoIT curriculum requirement for the Software Development track that states students in the software track must complete [redacted]

To support my case, I will cover five (5) different areas.

1. An overview of my background and my professional experience.
2. Five (5) reference letters from managers and peers at my current job.
3. A survey of some of the systems I work on or have worked on.
4. A discussion into some of the things I consider while on the job, mapped to the course's Learning Outcomes.
5. A list of certifications I hold.

Note: this is the only course I am using the PLA option for.

IT4045C Learning Outcomes

According to my Academic Advisor, [redacted], the course has seven (7) Learning Outcomes I must address in this support statement:

LO1. Describe the nature and scope of enterprise software applications.
LO2. Design distributed N-tier software application.
LO3. Research technologies available for the presentation, business and data tiers of an enterprise software application.
LO4. Design and build a database using an enterprise database system.
LO5. Develop components at the different tiers in an enterprise system.
LO6. Design and develop a multi-tier solution to a problem using technologies used in enterprise system.
LO7. Present software solution.

Section 1: Background and Professional Experience

I am a non-traditional student. In [redacted] I earned an associate degree from Cincinnati State in Business Computer Programming and Database Management. I am enrolled in the CECH [redacted]. I have a lot of professional experience in software development in an enterprise, but I want to finish my bachelor's degree to round out my academic background.

I have [redacted] years of professional experience in software development. Overall, I have been in the workforce for [redacted]

My professional experience includes the following:

- From [redacted] through present, I have been working for [redacted] in their "Technology & Operations Services" business line. I work on a team called "Intranet Development". We are a team that develops, maintains, and supports a portfolio of over 20 intranet-facing web applications and APIs, some of which are enterprise-wide. Our business line customers include Internal Corporate Communications, Human Resources, Community Banking, Elan Financial Services, Wealth Management / Investment Services, and more.
- From [redacted] to [redacted] I worked at [redacted] participating in the full SDLC lifecycle developing, maintaining, and supporting a product built for GE Aviation. This application modeled costs of engine parts over the life of military jet engines.
- In addition to my experience as a Software Developer in the full SDLC lifecycle, my experience also includes:
    - Acting as Lead Developer on numerous software development projects.
    - Business analysis to help create requirements, user profiles, use cases, agile user stories, acceptance criteria, and product backlogs.
    - Project management for small projects that didn't require a formal PMP-certified Project Manager from the Project Management Office.
    - Providing agile coaching and training including Scrum, XP, and Devops to our team and other teams.
    - 24-hour production incident on-call support.
    - Systems integration of vendor-built software onto company infrastructure.
    - Technical consulting for business lines as they explore vendor cloud-hosted SaaS solutions during discovery/requirements phase, and implementation phase. Implementations sometimes include coordinating SAML Federation configurations with InfoSec Specialists.
    - System service continuity planning and quarterly disaster recovery exercises.
    - Infrastructure updates and maintenance such as upgrading server Operating Systems and system migrations across data centers (e.g., move a system from 'Data Center A' to 'Data Center B').

Section 2: Letters of Reference

I am providing reference letters from peers and managers as validation of my professional experience in, and competence of, enterprise-grade system development. **Feel free to contact them via e-mail to verify their references.**

- The letters are separate PDF documents I uploaded with this application.
- **Please contact U.C. Testing Services if you are missing the five (5) PDF documents of the reference letters.**
- Refer to Table 1 below for details on each person's position, location, contact information, and how their reference maps to the course Learning Outcomes.

Table 1. Reference Letters

Letter	Person's Name	Position, Location	Learning Outcome Mapping	Org. Hierarchy	Dates	Contact Info
A	[redacted]	[redacted]	All 7	My Direct Manager	present	[redacted]
B	[redacted]	[redacted]	1,2,3, 5,6,7	Peer employee, mentor	present	[redacted]
C	[redacted]	[redacted]	4	Peer employee	present	[redacted]
D	[redacted]	[redacted]	All 7	My 2-up Manager	present	[redacted]
E	[redacted]	[redacted]	All 7	My 2-up Manager	Throug h	[redacted]

Section 3: Applications List

This section includes a *sample* of the systems that I work on or have worked on in the past [redacted] years at [redacted]

My role on each of these has been *at minimum* the full SDLC lifecycle. This includes designing, building, testing, deploying, and supporting new features and defect/bug fixes.

In some cases, I was present for the initial development. In other cases, I was added to the product later in its life to build enhancements and provide support.

I am providing 1-2 screenshots for each application I cover in this support statement. I work on internal systems, meaning that you would never see these systems as a bank customer. Due to the company's Information Classification policy (e.g., public, internal, confidential, etc.), I am not allowed to show some of the data on these screenshots to external readers. To prevent policy violation and potential termination, I blurred out areas of the provided screenshots that display potentially sensitive data.

Regarding the Learning Outcome #4, database development, compared to the other Learning Outcomes, I have less experience. We have a dedicated Senior Database Developer on our team. Most of the development work for larger systems with complex data models and complex queries is typically reserved for him.

However, I still have experience with [redacted] in building databases for some of these systems. When that happens, the Senior Database Developer will review and sign off on my work. My experience also includes times where I needed to reverse engineer databases and stored procedures to pinpoint a problem like a bug or defect.

InsideUSB / USBnetPortal
LO Mappings: 1,2,3,5,6

InsideUSB is a custom-built Content Management System (CMS) and rendering engine.

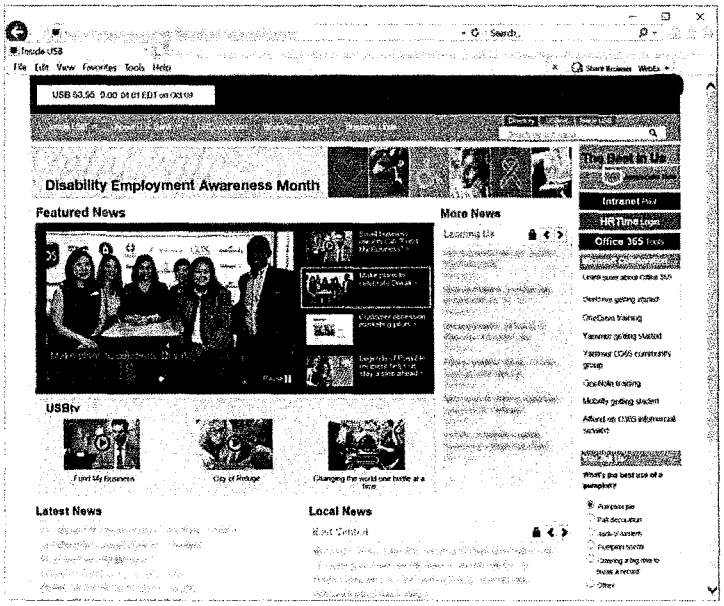
From an employee's perspective, this system acts as the internal news site for all 80,000+ employees. This system pulls together content that Internal Corporate Communications publishes and renders it. All bank internet browsers are configured to use this site as the browser's default home page instead of Google, Bing, or MSN. We average about 1 million pageviews each business day.

This system includes custom-built components including news articles, featured articles, news archives, polls, stock price ticker, searches, RSS feed reader, reader comments, and comment moderation management. Yes, we have had a few employees post unprofessional comments on the bank-wide site for all to see.

InsideUSB has a UI layer, business layer, and data layer. We also use some database caching mechanisms. We have updated it several times over the years to include new features and to do a full modern redesign. This system is integrated with an internally-hosted Google Search appliance for content searches. USBnetPortal is the same engine as InsideUSB but it's a system where other business lines can host their own intranet sites. See Figure 1 screenshot below.

My role: I am one of five developers who works on this system, mostly construction and production support.

Figure 1. InsideUSB front page



FIID (Financial Institution ID) Search
LO Mappings: All 7

The bank has a subsidiary called [redacted] They provide ATM and credit card products and services to their customers which are small banks and credit unions.

FIID Search is in-house built Customer Relationship Management System (CRM). The word FIID is the name for a customer identifier in Elan. Employees in networking and Elan use FIID Search to look up customer information. FIID Search stores customer information including basic customer account info, customer billing history, a customer's credit card portfolio information, and ATM machine network information.

Like our other systems, FIID Search has a multi-layer architecture distributed across many servers at our primary and backup data centers as well as a database. This system also uses a series of automated data import jobs from mainframe systems. See Figure 2 and Figure 3 screenshots on the next page.

My role: I am the only developer left who supports this system, as the other developer left the company. In recent years I was tasked with updating the system to work with a security middleware upgrade.

Figure 2. FIID Search page

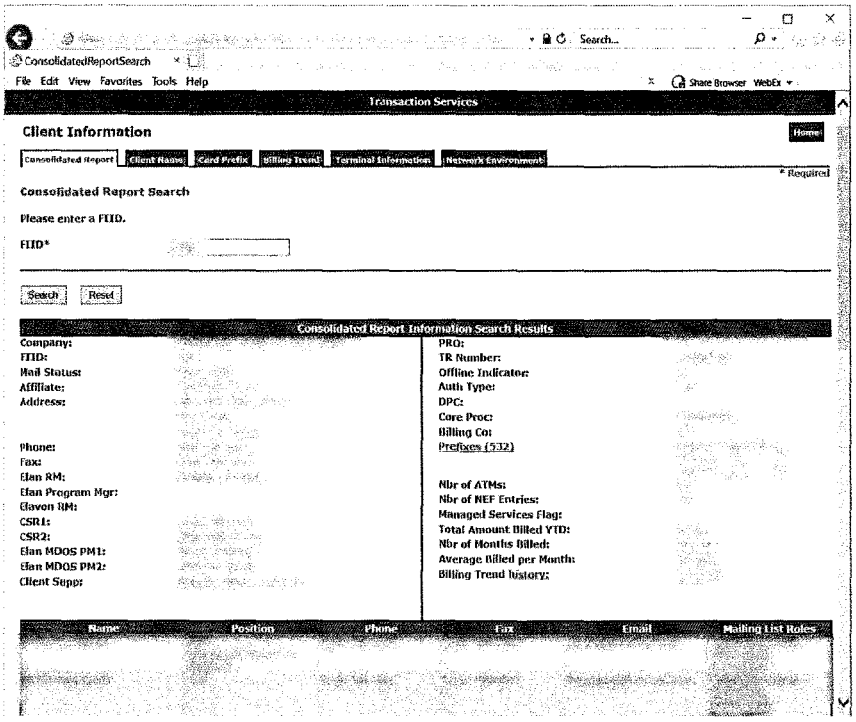
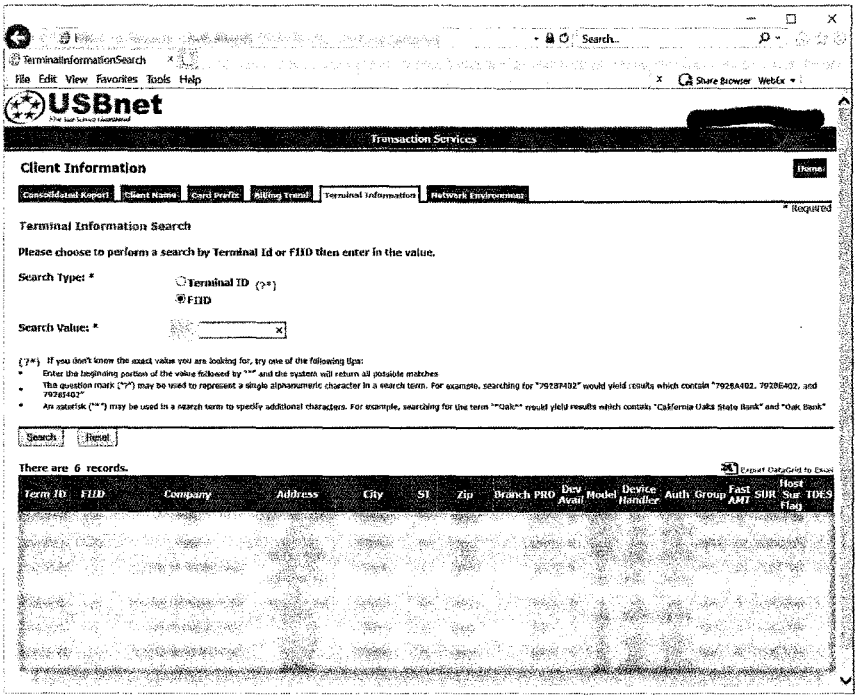


Figure 3. FIID Search, another page



OSDM (Operations Services Data Management System)
LO Mappings: 1,2,3,5,6,7

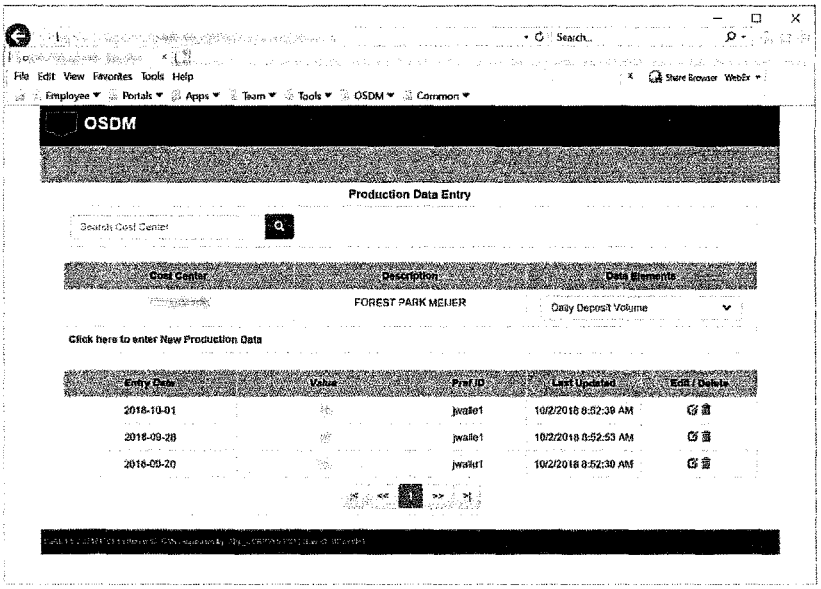
OSDM will be an in-house built Business Intelligence system with a Data Warehouse and Analytics component. We are currently building this system. We are early in an agile development effort iteratively building and deploying OSDM in increments to the bank.

Employees around the bank will have one place to enter daily and monthly data (e.g., number of transactions, checks processed, etc.). Analysts will use OSDM to track the various areas performance, set goals for them to work towards, and run reports for upper management.

Like other systems, OSDM will use a multi-layered architecture that is distributed across several web servers, database server, 2 data centers, and behind a load balancer. This system will also include several automated data imports and Tableau data reporting functionality. See Figure 4 screenshot below.

My role: I am one of 5 software developers building the system. I am also taking part in UX design, system design, construction, testing, and deployment. I am also serving as Scrum Master and "Release Manager".

Figure 4. OSDM data entry screen



AutoPayForms
LO Mappings: 1,2,3,5,6,7

This system is built for Branch Bankers (i.e. Tellers) across 4,000 branches.

Consumer Banking still needs to provide customers the ability to let them submit paper-based forms for billing Autopay change requests. Tellers were filling out billing AutoPay change requests by hand and making a lot of errors.

This system consists of a series of forms that Branch Bankers use to fill out customer billing "AutoPay" change requests online instead of paper. After the Teller is finished, the system generates PDF documents for the teller to submit to back office for processing. The system reduces the number of requests submitted with errors.

This system uses a 2-tier architecture consisting of UI layer and business logic layer. It also has a component for PDF report generation. The business logic layer holds the business rules and autopay request validations, ensuring the Branch Banker fills it out correctly. It does not require a database because we are not storing any data.

We explored and recommended a system design where the request would get submitted electronically to the correct back office area for processing, but the business line did not want to pursue at the time of development. See Figure 5 and Figure 6 screenshots below.

My role: This is a system I built by myself since it is a smaller system.

Figure 5. AutoPayForms input form

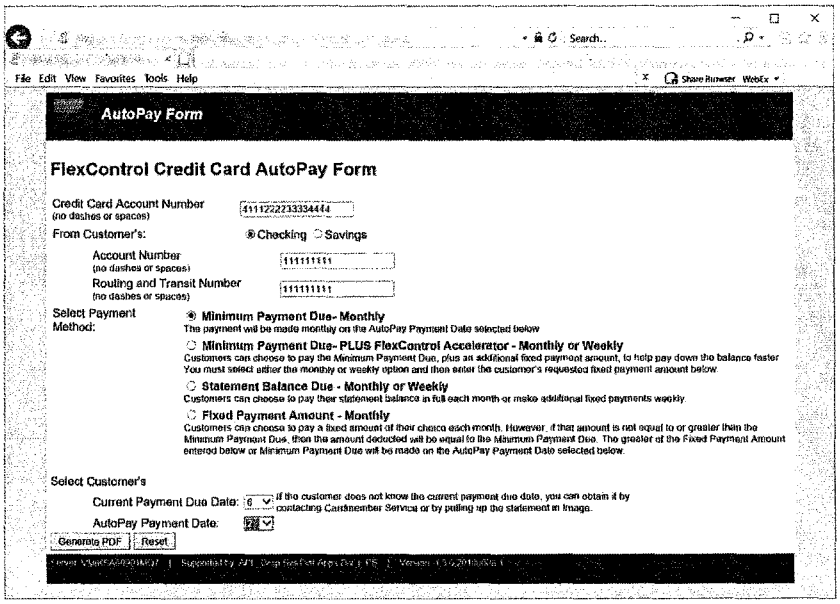
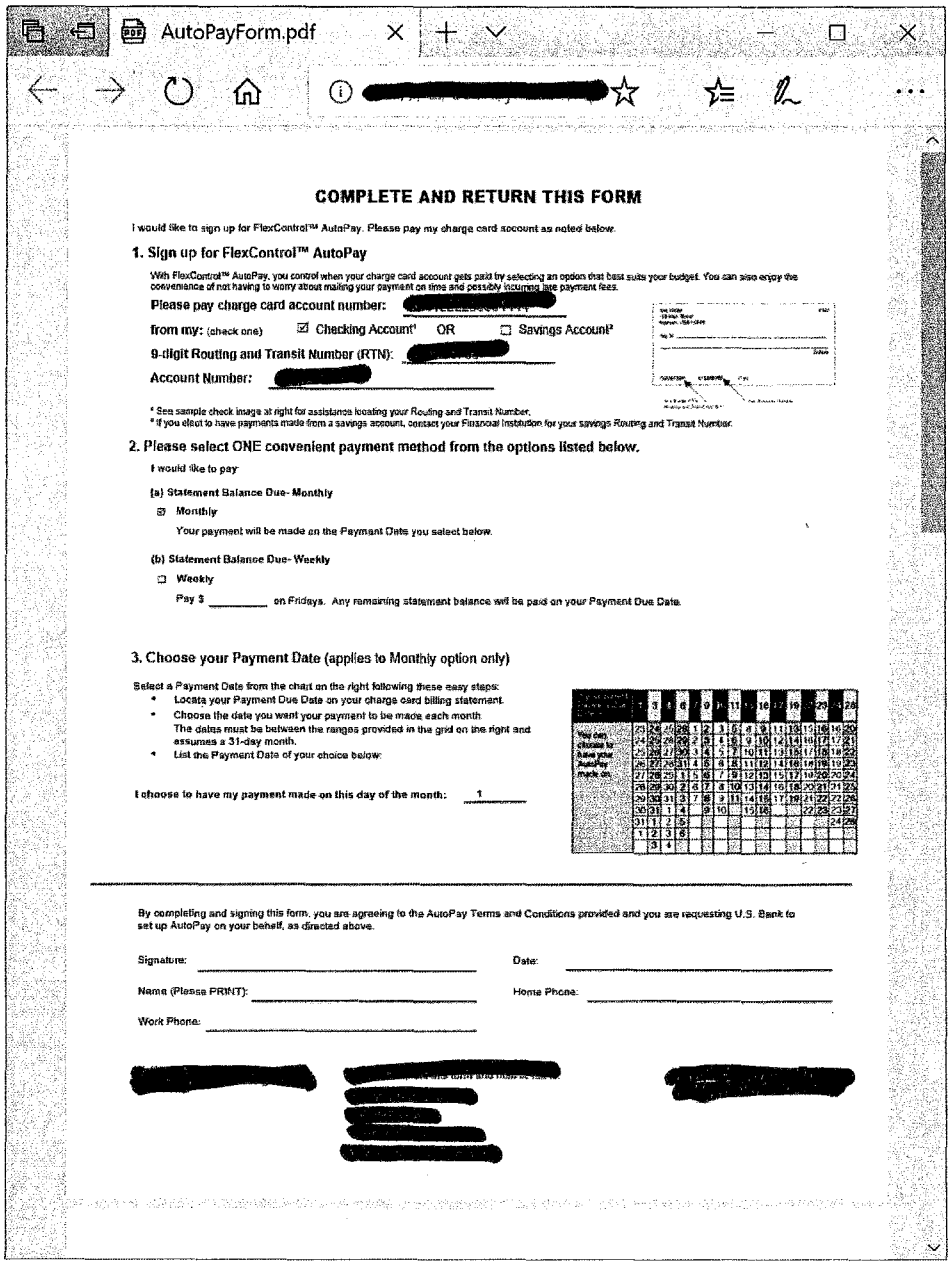


Figure 6. AutoPayForms PDF output



AutoPayForm.pdf

COMPLETE AND RETURN THIS FORM

I would like to sign up for FlexControl™ AutoPay. Please pay my charge card account as noted below.

1. Sign up for FlexControl™ AutoPay

With FlexControl™ AutoPay, you control when your charge card account gets paid by selecting an option that best suits your budget. You can also enjoy the convenience of not having to worry about mailing your payment on time and possibly incurring late payment fees.

Please pay charge card account number: [redacted]

from my: (check one) ☑ Checking Account[1] OR ☐ Savings Account[2]

9-digit Routing and Transit Number (RTN): [redacted]

Account Number: [redacted]

[redacted]

[1] See sample check image at right for assistance locating your Routing and Transit Number.
[2] If you elect to have payments made from a savings account, contact your Financial Institution for your savings Routing and Transit Number.

2. Please select ONE convenient payment method from the options listed below.

I would like to pay:

(a) Statement Balance Due- Monthly

☑ **Monthly**

Your payment will be made on the Payment Date you select below.

(b) Statement Balance Due- Weekly

☐ **Weekly**

Pay $ _____ on Fridays. Any remaining statement balance will be paid on your Payment Due Date.

3. Choose your Payment Date (applies to Monthly option only)

Select a Payment Date from the chart on the right following these easy steps:

- Locate your Payment Due Date on your charge card billing statement.
- Choose the date you want your payment to be made each month. The dates must be between the ranges provided in the grid on the right and assumes a 31-day month.
- List the Payment Date of your choice below:

I choose to have my payment made on this day of the month: 1

You can choose to have your AutoPay made on:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28
	25	26	27	28	1	2	3	5	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
	24	25	26	29	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	17	18	19	20	21	22	23	24
	25	26	27	30	3	4	5	7	7	8	9	10	11	12	13	14	15	16	17	18	18	19	20	21	22	23	24	25
	26	27	28	31	4	5	6	8	8	9	10	11	12	13	14	15	16	17	18	19	19	20	21	22	23	24	25	26
	27	28	29	1	5	6	7	9	9	10	11	12	13	14	15	16	17	18	19	20	20	21	22	23	24	25	26	27
	28	29	30	2	6	7	8	10	10	11	12	13	14	15	16	17	18	19	20	21	21	22	23	24	25	26	27	28
	29	30	31	3	7	8	9	11	11	12	13	14	15	16	17	18	19	20	21	22	22	23	24	25	26	27	28	29
	30	31	1	4		9	10		12	13	14	15	16	17	18	19	20	21	22	23	23	24	25	26	27	28	29	30
	31	1	2	5																								
	1	2	3	6																								
		3	4																									

By completing and signing this form, you are agreeing to the AutoPay Terms and Conditions provided and you are requesting U.S. Bank to set up AutoPay on your behalf, as directed above.

Signature: _____ **Date:** _____

Name (Please PRINT): _____ **Home Phone:** _____

Work Phone: _____

[redacted]

WMIS Workflow Navigator
LO Mappings: All 7

This system is built for Wealth Management / Investment Services (WMIS) back office employees who process various requests submitted by Investment Advisors.

WMIS had an issue where the decision-making process employees had to navigate to figure out which forms to fill out and which processes to follow was too complex. In programming speak, there were layers upon layers of nested "if statements" for humans to figure out on their own. People would use documentation to help them but keeping everyone's documentation up-to-date was a challenge. As a result, there were a high number of incorrect steps being taken and incorrect forms being submitted.

We designed a "TurboTax for WMIS back office employees". Just like how TurboTax asks a taxpayer questions to determine how it will fill out tax forms, this system asks the employee a series of questions to help them find the right forms and processing steps.

The system is designed to only ask certain questions based on previous answers. An employee might see a question like "*is this request Fast Track? Y/N*" or "*is this request Behavioral Based? Y/N*". So, if you answer question "A" a certain way, you may not need to be asked questions "B" or "C". If you answer "yes" to question "D", you might get an entirely different set of following questions.

After the system gets enough information via answers, the system will output a list of the correct forms the employee needs to complete or steps they need to take. The system will also provide the employee valuable information like SLAs, processing times, and contact information for questions.

Just like our other systems, this system has a multi-layer architecture distributed across many servers at our primary and backup data centers. It also includes a database for lookup information. See Figure 7 and Figure 8 screenshots on the next page.

My role: I am one of two (2) software developers that built this system. I also handled the UX design, system design, database design, and stored procedures.

Figure 7. A WMIS Workflow Navigator question screen

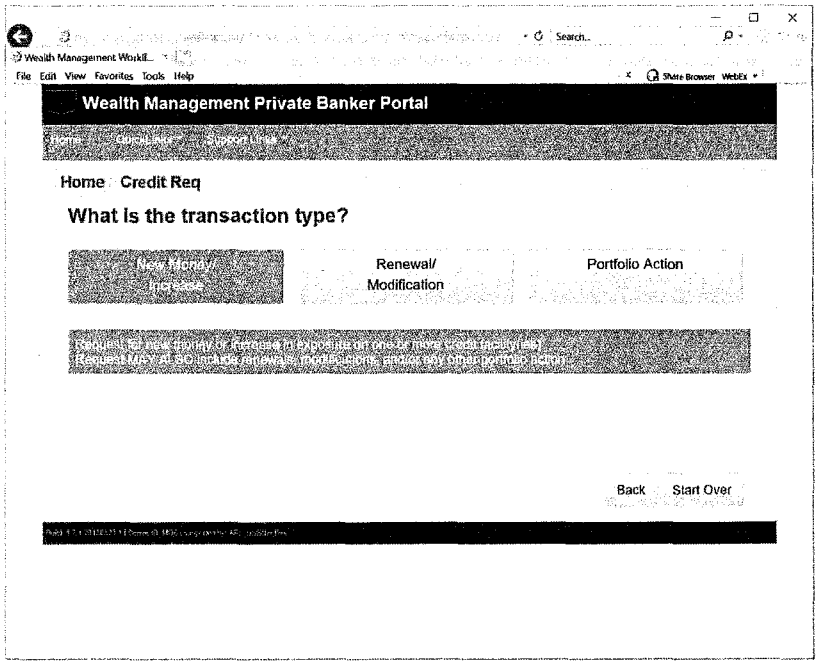
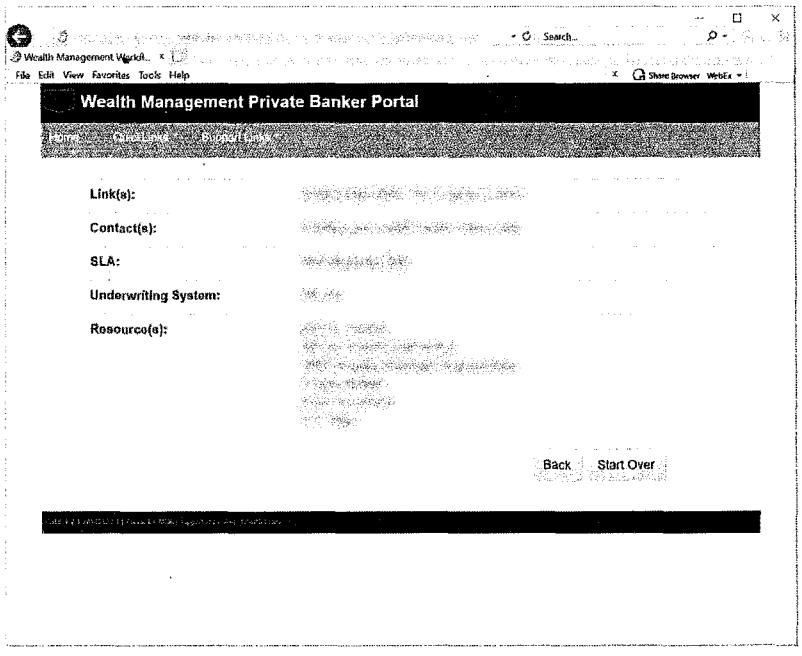


Figure 8. WMIS Workflow output screen



Section 4: Experience Mapped to Course Learning Outcomes

LO1- Describe the nature and scope of enterprise software applications

Here is a list of some of the pieces I have addressed in the past regarding the scope of enterprise systems design, development, and support:

- **System Architecture-** Although a lot of applications I work on share a common architecture, we make sure we determine the following:
    - How we want to break the system up into separate components much like how Lego pieces fit together.
    - How internal components within a system talk to each other via OOP Interfaces.
    - How multiple systems in different layers, or different physical locations talk to each other via some protocol like HTTPS.
    - How we host different components across a server infrastructure.
- Here are some of the types of components or "Lego pieces" I might address in systems I work with, depending on the business needs:
    - **Application layers-** within one "system" we may have a mobile app, a web app, and a desktop app that all need to use the same business rules and requirements. It's a bad practice to have a copy of the same business logic inside those each of the 3 examples. Instead we break out the business logic into 1 layer and have all 3 systems share it. A business logic layer will talk to a data accessor layer using shared Object Oriented Interfaces. The UI layers (web, mobile, desktop) might talk to an API via a RESTful HTTPS API usually or some other protocol.
    - **Class libraries-** (e.g., DLLs in .Net) Sometimes we will host a common set of helper libraries in a central location on a Windows Server for example in the Global Assembly Cache. The .NET Framework library is itself a large set of class libraries.
    - **Directory components**- for example Active Directory over LDAP.
    - **Middleware components-** for example, a security component might be installed inside IIS Server, and intercepts all web requests to determine if the request meets the rules for a security policy.
    - **Reporting**- Instead of reinventing the wheel, we can use a reporting system or component and just build custom reports based on that technology.
    - **Notifications**- We would never host our own email server. Instead we would interface with the company's email server using SMTP-related protocols or some SMS service.
    - **Federation**- while our internal systems don't need federation, we do act as Technical Owners of vendor-hosted systems. To enable an internal user to access it securely, we will use SAML Federation.

- We use **Object Oriented Programming** classes and design patterns primarily to help us build systems. Using Interfaces for example enables us to achieve loose coupling via Dependency Injection. That way, if we need to rewrite layer X, we won't have to change Layer Y to accommodate the changes in layer X. Plus, OOP helps us build applications that have objects based on the business needs.

Other areas I consider in scope for an enterprise software application include:

- **SDLC-** At the core, regardless of our method, we always go through the typical software development phases which include requirements gathering, design, construction, testing, deployment, and support.
- **SDLC Method-** In the past we would use a waterfall model. With waterfall, the SDLC phases would occur sequentially for 6-24 months before releasing it to the customer. Today however we operate in an agile culture using the Scrum framework. We focus on iterative development, where we develop and release smaller pieces to the customer many times to get feedback sooner. This allows us to validate that the solution meets their need, or pivot in case we are wrong. It's much better to fail small over a few weeks than to fail big over 6-24 months and find that we didn't solve the business problem. Agile helps us reduce the risk that a change (new system, update) will negatively affect broader business processes.
- **Code Reviews-** all development done at our company must be code reviewed by senior developers before it can be merged into the main branch.
- **Source Control-** Another area involves source control. We use Team Foundation Server for all our needs regarding GIT and TFVC (a git alternative) source control, code reviews, automated builds (continuous integration), and managing work with TFS Scrum tools (user stories, Kanban board, etc.). The bank requires our code to be stored in a central source code repository and not on developers' workstations.
- **Automated and Manual Testing-** We use XUnit and automated test runners for most of our unit testing. For QA testing, we create test plans including regression tests and run them manually. We also use static security testing tools like HP Fortify.
- **Performance-** We must account for system performance under a heavy load of simultaneous users. We use several components for this. We use several servers, not just one, behind a load balancer to eliminate one server getting hammered and slowing to a crawl. We also use various caching techniques between browser-server, and web server-database to reduce to number of network calls to the web server and database. We also perform load testing, using automated tools that come with Visual Studio IDE. We stress the system with heavy load to see when it fails. Then we analyze where we can improve performance in the system.

- **Security**- We must account for security of enterprise systems. For example, we use OWASP Top 10 as a framework for defending our intranet applications against SQL injection, cross-site scripting attacks, among other attacks. For example, when a post to the server contains any form data, we match it against a white list of acceptable values, along with HTML encoding it to prevent scripts getting saved to the database. We are also migrating a lot of our legacy systems over to HTTPS everywhere, with Certificates provided for by InfoSec.
- **Access Model / Role-Based Access Management**- We also must account for the type of access a user should have. Several systems or parts of systems are "locked down" to a small set of users, usually a small set of business line administrators. Others are open to all employees.
- **Infrastructure**- Infrastructure we use must be able to handle a heavy workload, so our database servers are typically installed on standalone physical servers that have high memory and several CPU cores. Our web servers and application servers typically require less horsepower, since we can spread load across multiple servers in a web farm.
- **Service Continuity**- High availability is another major concern. We have several data centers located around the country. We typically place our systems in two (2) data centers, with one marked as "PROD" (production) and one marked as "DR" (disaster recovery). Although, we are moving towards an Active-Active model where both data centers will actively process requests.
- **Documentation**- This includes a broad set of documentation needs. From service continuity documentation, system documentation, end user documentation, product support documentation, and System of Record documentation (e.g., CAR-a central repository list of active systems in the bank, SCM-a central repository list of servers in the bank).
- **Service Continuity Planning**- In addition to the Service Continuity technology, we also do Service Continuity Planning. In the event of some service outage (data center outage, data center unavailable, etc.), employees manually step through a Service Continuity Plan (SCP) we manage. SCPs include Recovery Time Objectives (RTO) to recover the system as fast as possible, and Recovery Point Objectives (RPO) to minimize data loss as much as possible.
- **Production Support** and **Knowledge Management**- We must account for production support. To do this we make sure we create Knowledge Documentation for our Level 1 ITSC (IT Service Center) help desk, and Level 2 Production Command Center. We also recently added a dedicated Production Support team that we need to create documentation for and train on how to handle common issues with our systems. They handle more detailed system issues that Levels 1 and 2 cannot handle. We also work with the Production Support team to identify problems to reduce future incidents and outages.

- **Demand Management-** Before we begin building on a new system, from a big picture, all ideas of applications and systems must go through an Enterprise-wide process of prioritization. This is where the idea must be written down onto a business case outlining what the overall need/problem and desired outcome. If approved, this gets routed to the appropriate development team and placed on their backlog.
- **Change Management-** Since the bank has over 1,000 Application Developers and dozens of application teams, it is important that a large company has a defined set of guidelines, policies and governance regarding any introduction of change to the business. The bank uses ITIL. A change can include a new system, a system upgrade, a new network appliance, or a new business process. We use change management tools to document any changes we introduce to the business.
- **Regulatory Compliance-** Working in financial services technology, one area that we must account for is risk and regulatory compliance. We follow laws and regulations set by governments and industry groups, including Payment Card Industry rules (PCI), Sarbanes-Oxley Act of 2002 (SOX), the Basel Accords, and many others. Depending on the nature of the system, we have strict guidelines and steps we must take to engage Risk Management resources and get their guidance and approval at different points in a project.
- **Data Governance Controls-** One example of regulation we follow is the General Data Protection Regulation (GDPR) from Europe. We must be mindful how we collect, handle, and protect personal data. We must design and document how data goes through a data lifecycle. The data lifecycle we consider includes what type of personal data is being collected, how the data is collected, the user's consent, how data is transmitted, how data is stored, and when data is deleted.
- **Financial Management-** Most of our work falls under an operational "business-as-usual" BAU model. Occasionally though we will develop a new, large system, with development costs exceeding $1 million. In that case, we would use a "capex" project. This is where we collaborate with finance and senior leaders to plan what we want to build, determine a budget we need, get it approved by the Investment Committee, and then execute getting the solution built. The planning process alone can take up to a year and involves a PMP-certified Project Manager. This results in recording development and infrastructure costs as capitalizable expenses that get recorded as balance sheet assets that depreciate over several years, instead of recording it as operating expenses on the income statement.

LO2- Design distributed N-tier software application
Our system architecture typically consists of multiple tiers.

- UI layer (typically a web site or API)
- Business layer
- Persistence / data layer
- Business objects layer that all other layers inherit from
- Unit Test layer- not really a layer but it we have it as a separate "project" in a .Net system.

Our systems are distributed in that we have most systems installed on 3 PROD and 2 DR web servers, and databases installed on a dedicated PROD and DR servers. The web systems are placed "behind" a load balancer. We also have DEV, IT, and UAT environments we use for development, integration testing, QA testing, and user acceptance testing. See *Figure 9* below and *Figure 10* on the next page.

Figure 9. Typical application architecture

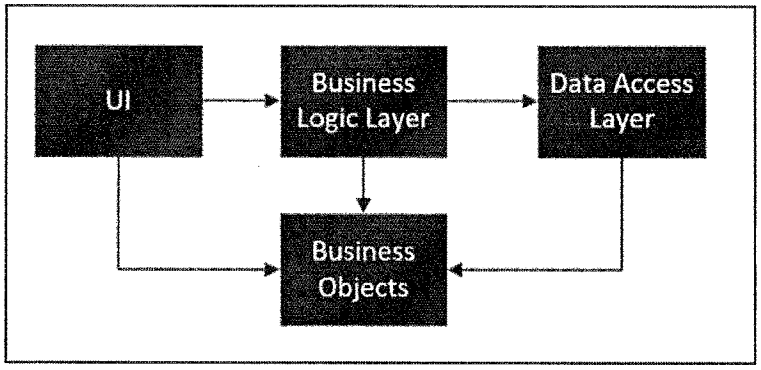
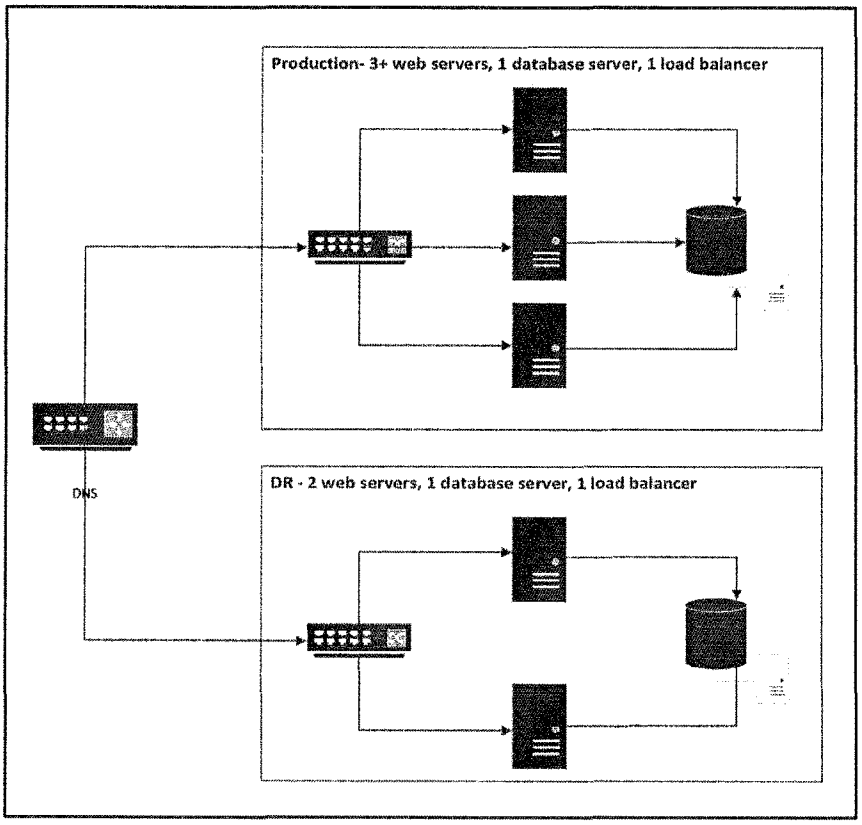


Figure 10. Typical system architecture



LO3- Research technologies available for the presentation, business and data tiers of an enterprise software application

My team is a ".Net shop" meaning our technologies are primarily focused on Microsoft technologies. The bank also has Java teams, but those are in different areas.

Enterprises like U.S. Bank need vendors that provide the technology we use, to provide strong service support plans. For example, Microsoft provides a support plan for SQL Server. Because of this need, we are not allowed to use "bleeding edge" major technology without it being vetted by the Enterprise Architecture Technology team. Vendors my employer prefers include Microsoft, IBM, SAP, Salesforce, Oracle, HP, CA Technologies and some others. Open-source technology is often discouraged unless it is a small component like a UI calendar plug-in.

As for this Learning Outcome:

- **Client-side** technologies- We utilize style libraries like bootstrap to give us robust, responsive layout options. We also use Jquery for simple JavaScript client-side code like validation, animations, and AJAX calls. At this layer, we use different techniques like bundling and minification to reduce the number of calls for static files. We also use client-side caching to reduce the number of calls for files like JavaScript files that don't change that often. We are however moving towards a model where we have the client handle all the UI "workload" using Angular. We would use the server to handle API calls from Angular, and of course deliver the static HTML, CSS, and JS files.
    - Currently, we do not use a "single page application" (SPA) design where most of the heavy lifting happens on the client-side with Angular or React talking to server-side APIs. But that is something we are considering.
- **Server-side presentation layer** technologies- We use C#, ASP.Net MVC, or ASP.Net WebForms for legacy systems. We use this when we want to either create RESTful Web APIs for JSON data responses, or web applications where the server will render dynamic HTML content. Currently, we are using .Net Framework, but we are moving towards the new .Net Core framework. Some of our legacy services also include SOA-based Microsoft WCF components.
- **Business logic layer** technologies- While we mostly use this for business rules, we sometimes introduce data caching to store commonly accessed data into a session or local cache. This reduces the number of round trips to the database, thus improving system response time performance. Another technology for database caching that we've researched but we currently don't use is Reddis.
- **Data accessor layer** technologies- We either use Entity Framework or plain ADO.Net libraries to call into the database. The direction we choose depends on how complex the call to the database is. For simple CRUD calls, we use EF but for more complex calls to the database, where EF might slow the system down, we use ADO.Net.

- **Data persistence tier** technologies- We typically use SQL Server relational databases for our data store layers. Our current data needs involve OLTP transactional type datastore as opposed to an OLAP data warehouse solution. We have investigated MongoDB, a NoSQL solution, recently during technology research.
- **Authentication** and **authorization** technologies- We are mandated by Information Security policy to use a common middleware component called "WebGate" by Oracle. InfoSec manages the system-specific policies that dictate which Active Directory domain groups have access to certain resources. They manage that using a tool called Oracle Access Manager. For authentication, since we are solely focused on intranet applications, we use the built-in "integrated windows authentication" feature in the IIS Web Server to help us confirm the user is logged into the bank network. We rarely link directly to Active Directory over LDAP anymore, per guidance from the networking group.

Other considerations include:

- **Unit testing** technologies- We use XUnit for our unit tests.
- **Quality assurance testing** technologies- We use TFS Test Suites filled with Test plans and to conduct manual testing.
- **Reporting** technologies- We either use a ReportViewer component from Microsoft for simple reporting needs inside the system, or for more robust reporting needs, we use either WebFocus, SSRS (SQL Server Reporting Services), or Tableau. Lately, Tableau has been the go-to solution and a favorite among business line customers. I have built some basic ReportViewer reports. I am still training on Tableau reporting.
- **Job automation** technologies- Many of our bank systems will either import nightly feeds or export feeds, in the form of flat files. To make this work, we use a job runner called Autosys, that executes an ETL job. The jobs are typically SSIS (SQL Server Integration Services) packages along with some stored procedures. For imports, the SSIS package will find the file out on a share drive and import it into a database our system uses. For exports, the SSIS package will create a file and place it on some share for another system in the bank to consume. I have built a few basic SSIS packages along with stored procedures and Autosys schedules.
- **Hosting** technologies- many companies consider whether to host their systems on-premise or on an external cloud provider like Azure or AWS. We currently host internally on-premise only. However, I have heard rumors we might change our hosting model in 3 to 5 years.

LO4- Design and build a database using an enterprise database system
All my database work involves relational databases. My work is reviewed and approved by our team's senior database developer. My work is also reviewed by the company Database Administrators who check for best practices and coding standards before they approve a deployment to our UAT and Production environments.

All database work must be designed with 3NF. If we don't use Entity Framework Core, we must use Stored Procedures. In some instances, we use SQL data caching for high volume database traffic. For some scenarios, we use automated data import and data export jobs for data feeds and extracts.

LO5- Develop components at the different tiers in an enterprise system
LO6- Design and develop a multi-tier solution to a problem using technologies used in enterprise system
Since these Learning Outcomes are closely related, I will combine them.

Please refer to *Section 1: Background and Professional Experience* and *Section 3: Applications List* to review my work experience. Refer to *Section 2: Letters of Reference* for validation from my co-workers.

My work experience helps makes a case that I have real-world experience designing and developing components and system layers, combining into full enterprise-grade multi-tier systems that can scale, and doing all of it to solve business problems covering small business units up to the entire enterprise.

LO7- Present software solution
In terms of presenting, this is one area I really enjoy. I have given many presentations including stakeholder reviews, system demos, and end-user training. One recent trend I have been taking part in is producing short YouTube-like videos that I publish on the intranet, share the link, and employees can watch on their own time.

I do some technical writing, mostly in the form of either system documentation to help us keep track of system information or "how to" documents to help us work through the bank's many processes we must follow.

Section 5: Industry Certifications

Refer to Table 2 below.

Table 2. Industry Certifications

Certification	Full Name	Agency	Supporting Link
PSD	Professional Scrum Developer 1	scrum.org	[redacted]
PSM	Professional Scrum Master 1	scrum.org	[redacted]
PSPO	Professional Scrum Product Owner 1	scrum.org	[redacted]

Summary

My goal was to make a case that I have real-world experience that aligns with the Learning Outcomes of the course [redacted] I hope this support statement describing my experience and background, along with the reference letters, captures that.

If you have any questions, feel free to contact me via email or phone. I am also happy to meet via Webex or in person if needed.

Thank you for your consideration.

[redacted]

Table of Figures and Tables

[redacted]

To whom it may concern,

It's my pleasure to write to you regarding [redacted]oftware development experiences with [redacted] and how they relate to the [redacted] course at UC.

[redacted] has been with [redacted] fo[redacted]years occupying various roles as his career has progressed successfully delivering numerous development projects ranging in orientation from divisional to enterprise in nature.

The various Line of Business division oriented solutions support ranging from 2k to 26k registered users with concurrent users ranging from approximately 1K to 3k. The Enterprise Intranet solution supports approximately 82k FTE's and contractors with concurrent users averaging approximately 6k users.

Solutions range in nature including:

- HRIS
- Intranet managed code solutions with vendor CMS integration
- Assorted back office and employee systems oriented solutions
- Compliance and regulatory
- Cloud SaaS federated solutions with custom integrations

Managed Code solution highlights:

- .Net C# and ASP.Net
- SQL Server 2008/2012/2014
- N-logical tier design with 2-physical tier design
- Scale-out and scale-up web tier, application tier, and data tier approaches
- OOAD
- SOA
- TPS - minimum 3NF schema design; Data mart – denormalization as appropriate
- Application design patterns satisfy automated security scan best practices

Primary roles occupied over tenure:

- Application developer
- Database developer
- Development lead
- Architecture lead
- Scrum master / Agile coach

Secondary roles:

- Business Analyst
- Test lead
- Infrastructure lead

[redacted] applied technical application design and development competencies are well respected by senior career IT professionals within the [redacted] Employee Systems division.

Thank you for taking the time to read my personal views regarding [redacted]; they do not represent an official statement from [redacted].

Best regards,

[redacted]

To Whom It May Concern:

In my role as Application Architect with [redacted] in the Intranet Development group, I have worked closely with [redacted] over the last [redacted] years. Our group is responsible for developing internally facing enterprise applications. Dependent on the business requirement the applications are utilized by employees and contractors numbering from the hundreds to around 90,000.

Our software is developed using Object Oriented Analysis and Design focusing on the principles of encapsulation, inheritance and polymorphism. All applications follow the same 5-layer (client-side scripting, UI code behind, Business Logic, Data Access and Business Objects Layers) 3-tier (Presentation, Application and Data tiers) design utilizing mainly the Microsoft stack of technologies. When appropriate, the layers are encapsulated within C# class libraries to promote reuse. The data is stored in SQL Server databases in third normal form and is accessed via ADO.Net utilizing stored procedures.

[redacted] has also been part of the design, development and implementation of base class libraries used to promote the standardization of data access, object serialization, logging and instrumentation.

The Microsoft technologies we have utilized include (but not limited to):

- IIS web server
- Microsoft SQL Server database (T-SQL)
- ASP.NET web forms (C#)
- ASP.NET MVC (C#)
- JavaScript, jQuery, and jQuery UI
- Bootstrap
- TFS for work management (both Waterfall and Agile approaches), source control, builds, and testing.

During [redacted] time, [redacted] has been involved in the design, coding (all tiers), testing and deployment of over 10+ applications.

Sincerely,

[redacted]

To Whom it May Concern,

I am the leader of a team of technology professionals at [redacted]. Our team supports enterprise workforce management solutions including custom developed applications, vendor package solution and cloud software.

[redacted] is a developer in my organization. [redacted] responsibilities include:

- Leadership, Communication & Collaboration: [redacted] is responsible for working with customers to define business needs, assess business value and justification, and manage escalation where required.
- Development: [redacted] is one of our senior .NET developers, both creating and maintaining existing applications. As part of [redacted] responsibility, [redacted] is responsible for:
    - secure code development following [redacted] standards including using scan tools such as Fortify
    - detail design and development of data, presentation and application layers
    - build integrated solutions using federated sign on, application monitoring tools, web services
    - Web based application design designed using role based access models (e.g. administrators, users, budget owners, etc.)
    - Participate in enterprise technology reviews, audit and assessments of application, database and architecture for platforms he developed or maintained.
- Employee Development: [redacted] has participated in .NET communities of practices, completes code review for other developers and continues to provide design consulting to other developers.

Please let me know if you have any questions or would like to discuss further.

[redacted]

To whom it may concern,

My name is [redacted]. From [redacted] when we hired [redacted], to [redacted], I was the Group Application Manager overseeing 40 people including [redacted] and his direct manager [redacted], before I moved to a different area at [redacted]

I reviewed the seven (7) required learning outcomes for this Enterprise App Development course at UC. I can confirm that in my time working with [redacted] he meets the requirements. He has the professional experience I think merits this PLA portfolio application he is seeking.

- [redacted] has worked on several projects involving end-to-end development of enterprise-grade systems.
- When he first started at the bank, he served as an individual contributor in a development team.
- Later, [redacted] served as a Lead Developer overseeing the development effort of a small team on smaller projects.
- [redacted] manager has informed me that since [redacted] has been promoted two times from Junior level developer, up to a more mid/senior level developer.

Overall [redacted] does have the professional experience here at [redacted] that I feel comfortable supporting his effort in this PLA application for [redacted]

Thank you for your consideration in this matter.

[redacted]