XYZ Dad Inc.

Customer Relationship Management Interface

Theodore Hickman

Version 1.0

CONTENTS

A. INTRODUCTION

The proposed Contact Relationship Management system is an HTML 5 based web application with an Oracle Database Management System backend. It has multiple integrations to provide contact management, sales tracking, contract management and order management to end users of American Video Game Company (AVG) via an administrative control panel and via a user control panel for third party vendors and affiliates. This proposal is designed to meet the requirements presented in the CRM Requirements document provided by AVG.

A.1. PURPOSE STATEMENT

The purpose of this document is to propose an efficient, scalable and compatible CRM solution for American Video Game Company (AVG). This will entail detailing an Introduction section, Requirements section, Software Development Methodology section, Design section and a Testing section.

Starting with the Introduction section we will layout an overview of the problem, the goals and objectives, prerequisites, scope and environment of the proposed solution. Next, the Requirements section will discuss the type of requirements necessary (business, user, functional, and non-functional requirements). From there we will detail the Software Methodology we are proposing and it's advantages. Following that we will propose an overview of the design proposed as well as a few design elements (UML, flowchart etc.) We will wrap up with the Testing section and some examples of testing we will be incorporating into the process.

A.2. OVERVIEW OF THE PROBLEM

Due to rapid growth and expansion, American Video Game Company has outgrown it's current Customer Relationship Management (CRM) system. Increased sales and therefore increased CRM resources, have highlighted the current CRM's inability to scale at the pace the company is growing. Due to a mixture of manual and automated processes as well as lack of integration between components of the CRM, the CRM has proven to be inefficient.

A.3. GOALS AND OBJECTIVES

The three goals of the proposed solution are efficiency, scalability and compatibility. The primary goal is to provide a Customer Relationship Management (CRM) system that efficiently manages current customer base. The second goal is to provide a solution that is horizontally scalable with expected growth. The final goal is to provide a CRM that delivers compatible artifacts and results in relation to the current CRM process.

These goals will be realized through the following objectives:

- Consolidation of contact and business information
- Reporting on client's interactions
- Activity Reports and Management
- Sales Tracking
- Data Integration with internal and third party systems

- Robust security
- Role based security for internal and remote resources
- Enhancements and increased scalability to presentation, logical and data layers.
- Access for third party marketing companies with contractual obligations

A.4. PREREQUISITES

There are three prerequisites necessary before the project can commence. The first prerequisite is a cloud based hosting solution, our recommendation is Amazon Web Services. Secondly there is a necessary production environment with Python 3.x and Django web framework. Lastly, Oracle Database Management System must be installed to facilitate the data storage and manipulation.

Number	Prerequisite	Description	Completion Date
1	Amazon Web Services (AWS) cloud based hosting service	Cloud based hosting service is necessary due AVG's need for integration across multiple offices on a national scale as well as horizontal scalability consistent with AVG's anticipated growth. Amazon Web Services will allow for the company to grow based upon it's needs with AWS' ability to add servers horizontally on-the-fly and AWS' ability to burst during high traffic times assuring their 5400 concurrent connections are met with timely results.	July 20 2020
2	Python 3.x with Django web Framework	Python 3.x is required due to the flexibility the language offers in design, development and testing frameworks. Python will be used for the Application Programming Interface and back-end integrations. Django is an industry standard web framework that supports templating for the presentation layer, integration with Oracle for the data layer and functional and unit testing frameworks to ensure efficiency and reliability.	August 1 2020
3	Oracle DBMS	Oracle is a proven leader in relational databases necessary to store all contact, sales and marketing information. Oracle has proven it can scale at the rates at which AVG projects. Oracle is necessary to maintain the level of reliability that AVG requires. Oracle's detailed support will enable AVG to mitigate any challenges with scaling and/or efficiency with their growth.	August 1 2020

A.5. SCOPE

This proposal will address the broad categories from the CRM Requirements Document (CRMD). The proposal will provide

The first requirement category this proposal will address is the Hosting Requirement CRMD. This proposal will not cover this requirement and instead offers to acquire hosting on the AVG's behalf

from Amazon Web Services. The hosting requirements of "demonstration of how connectivity outages, service level agreements (SLAs) and upgrades" of operating system and network level requirements will be provided by Amazon Web Services.

The next requirement category this proposal will address is the Users Requirement CRMD. This proposal will provide a scalability factor of 10,000 total users with a burst rate of 750 concurrent users. This proposal will ONLY provide a guarantee on the application and it's sub-parts. Network and operating system requirements will be provided by Amazon Web Services.

The next set of requirements in scope will be referred to as the Application Requirements (App). The App will provide a Contact Management section with the entirety of the business rules enumerated in the CRMD (Section Contact Management). The App will also provide the full requirements per section labeled Ticketing System in CRMD. Data types listed in section Data Types will be fully supported in both the App and the database. Reporting as described in the CRMD Reporting Section is not within scope of this proposal and instead we recommend Flashy Reports integration with the App (this is a third party software that has proven itself in the industry and integrates with the App seamlessly).

A.6. ENVIRONMENT

The environment for the proposed solution is composed of presentation, logical and data segments.

The presentation layer will be comprised of sixteen (16) web servers which will serve the user interface to the proposed CRM. This will include all static images, dynamic content and input forms.

The logical layer will be comprised of four (4) Application Programming Interface (API) servers that will serve API and web service requests for both internal and third party integrations. Only the logical segment will be able to access the data segment per security requirements.

The data layer will be comprised of eight (8) database virtual servers that will be utilizing Oracle DBMS Enterprise Edition to store and serve data. The data segment will only be in communication with the API layer of the proposed solution due to security constraints.

A detailed summary of the environment is included below:

The presentation segment will be made up of a front-end environment consisting of the following:

- Browser support will include:
  - Chrome
  - Firefox
  - Safari
- Mobile support
  - HTML 5 app
  - Support for mobile via website

Browser support will not include support for Flash or other third party software. It will strictly be based on HTML 5 standards with AJAX.

- Hardware:
  - Sixteen (16) web servers with the following configurations:
    - 2 TB SSD
    - AWS premium Virtual Machine
    - Linux 2.6
    - Python 3.x with Django web framework
    - EC2 storage to share data across web servers
    - Closed network to API servers

The logical segment will include a back-end and business logic elements consisting of the following:

Back-end elements:

- Hardware:
  - Four (4) API servers
    - AWS premium Virtual Machine
    - Linux 2.6
    - Python 3.x with Django framework
    - Closed network to Oracle

The data segment will include Oracle DBMS housed on it's own independent network with access to the API servers.

- Hardware:
  - Eight (8) Database servers
    - 4TB SSD storage
    - Linux 2.6
    - Oracle Database Management System Enterprise Edition
    - Closed network to API

B. REQUIREMENTS

This proposal will now address the business requirement of Forecasting Sales, the user requirement of Contract Management, the functional requirements of Soft Delete and Historical Archive and the non-functional requirement of User Growth and Scalability.

B.1. BUSINESS REQUIREMENTS

Forecasting Sales is a business requirement that this proposed solution will address by providing integration to a business logic layer and storage of previous sales and revenue to predict future revenue and sales utilizing proprietary algorithms.

The information compiled by the business logic layer will be available in the administrative control panel for management to access.

B.2. USER REQUIREMENTS

Contract Management is a user requirement that will be accomplished through a separate module in the administrative control panel. Contracts and their metadata will be storage in the Oracle DBMS. Users will be able to create, modify, approve and terminate a contract within the administrative control panel.

B.3. FUNCTIONAL REQUIREMENTS

Soft Delete is a functional requirement that will be addressed by adding a “viewable” Boolean field to each table in the database. Only items that have the visible flag set will be viewable in the user interface. When a deletion is executed this flag will be turned off (set to false) for the item that has been deleted.

Historical Archive is a functional requirement that will be implemented via an SQL trigger to copy the record of a table that is deleted to a historical table. The date, time, created by and deleted by will be added to each record in the historical table in order to preserve the history of table in an event of an audit.

B.4. NONFUNCTIONAL REQUIREMENTS

The User Growth and Scalability requirement is a non-functional requirement that concerns the number of overall users as well as concurrent users the proposed system should support. This proposal will accommodate this requirement by using a scalable Oracle Database Management System coupled with Amazon Web Service which will allow the concurrent connections to burst during peak times supporting up to 750 concurrent users. Additional servers will be added on-the-fly when users have reached 80% of the threshold of 500 concurrent users.

C. SOFTWARE DEVELOPMENT METHODOLOGY

Two software development methodologies were chosen as candidates for this proposal. After reviewing the predictive Waterfall Software Methodology and adaptive Agile Software Methodology we found the best methodology to incorporate the needs and requirements of the project to be Waterfall as detailed in this section.

C.1. ADVANTAGES OF THE WATERFALL METHOD

The advantages of the Waterfall software development methodology according to UCertify (2017) are:

- **Predictability**—If everything goes according to plan then you know exactly when different stages will occur. In particular, you know when you'll be finished and how much effort (aka money) you'll need.
- **Stability**—Because the requirements are "set in stone" at the beginning of the project, the customers know exactly what they are getting..
- **Cost-savings**—If the design is clear and correct, you won't waste time following development paths that turn out to be dead ends.
- **Detailed design**—If you design everything correctly up front, then you shouldn't need to waste time making a lot of decisions during later development. You just follow the plan. That makes programming faster (and therefore cheaper).
- **Less refactoring**—Adaptive projects tend to require refactoring. A programmer writes some code. Sometime later, the requirements change and the code needs to be modified. The result may need to be refactored to make it more efficient or to satisfy code standards. These problems don't occur as often in predictive projects.
- **Fix bugs early**—If the requirements and design are correct and complete, then you won't have to fix any bugs they would have caused later. Because it's easier to fix bugs early on, that saves you more time.
- **Better documentation**—Predictive models require a lot of documentation before programming starts, so you at least have some documentation.
- **Easy maintenance**—Because you can consider the application from a broader perspective, you can create a more elegant design that's more consistent and easier to maintain.

C.2. DISADVANTAGES OF THE WATERFALL METHOD

The disadvantages of the Waterfall software development methodology according to UCertify (2017) are:

- **Inflexible**—Just because you *thought* the requirements wouldn't change, that doesn't mean they won't. If they do, accommodating them can be hard. (Basically, a predictive model is a gamble that requirements won't change too much.) That lack of flexibility also means you can't take advantage of new opportunities. If someone invents a new easier way to display sales reports, you won't be able to take advantage of it.
- **Later initial release**—Many adaptive models enable you to give the users a program as soon as it does something useful. With a predictive model, the users don't get anything until development is finished.
- **Big Design Up Front (BDUF)**—You need to define everything up front. You can't start development until you know everything you're going to need to do.

C.3. ADVANTAGES OF AGILE METHOD

The advantages of the Agile Software Development Methodology according to UCertify (2017).are:

- More accurate requirements. The customers can adjust the requirements as needed during the project.
- The ability to track changing requirements. If requirements must change (within reason), the project can start tracking the new requirements in the next iteration.
- Frequent customer feedback and involvement. In addition to helping keep the project on track, this keeps the users engaged with the project.
- Reduced development time. If everything goes smoothly, you don't spend as much time writing requirements in excessive detail.
- Encourages code reuse. One of the key RAD ideas is to do whatever it takes to get the current iteration done. If an existing piece of code does what you need it to do (or even almost what you need it to do), timeboxing encourages you to use that code instead of writing something new.
- Possible early releases with limited functionality.
- Constant testing promotes high-quality code and eases integration issues.
- Risk mitigation. Before each iteration, you can look for potential risks and handle them.

- Greater chance of success. BDUF projects sometimes spend a lot of time following an incorrect path before discovering they're heading in the wrong direction and they need to be radically redone or even canceled. Frequent increments allow RAD projects to detect and correct problems quickly before they become insurmountable.

C.4. DISADVANTAGES OF AGILE METHOD

The disadvantages of the Agile Software Development Methodology according to UCertify (2017). are:

- Resistance to change. It can be hard to get existing software engineering groups to adopt new RAD models, particularly given how odd some of their techniques can seem.
- Doesn't handle large systems well. Big systems require a lot of effort, and that usually means a lot of people.
- Requires more skilled team members.
- Requires access to scarce resources.
- Adds extra overhead if the requirements are known completely and correctly in advance.
- Less managerial control. Many managers have trouble allowing a project to head off in its own ever-changing direction.
- Sometimes results in a less than optimal design.
- Unpredictability. Some customers just want to know how much and how long, and they really aren't interested in shaping the application throughout its development.

C.5. BEST SUITED

Waterfall Software Development Methodology is best suited for this proposal due to the detailed nature of the requirements produced by AVG, the customer's desire for maintainability and the need for extensive documentation early in the design phase. A further benefit is the ability to project costs based on detailed high-level and low-level design early in the project. The goal of this project is to produce software with all of the required features with high fidelity, this option is not possible with a more iterative approach like Agile Software Development Methodology.

D. DESIGN

Included below is high-level design of the requirement to use AVG's ActiveDirectory as well as a high-level network design to facilitate the necessary scaling of the application via Amazon Web Services (AWS).

D.1. FLOWCHART

The following is a flowchart diagram detailing how the proposed solution will facilitate the requirement of logging in users via AVG's ActiveDirectory as well as the requirement to store login history:

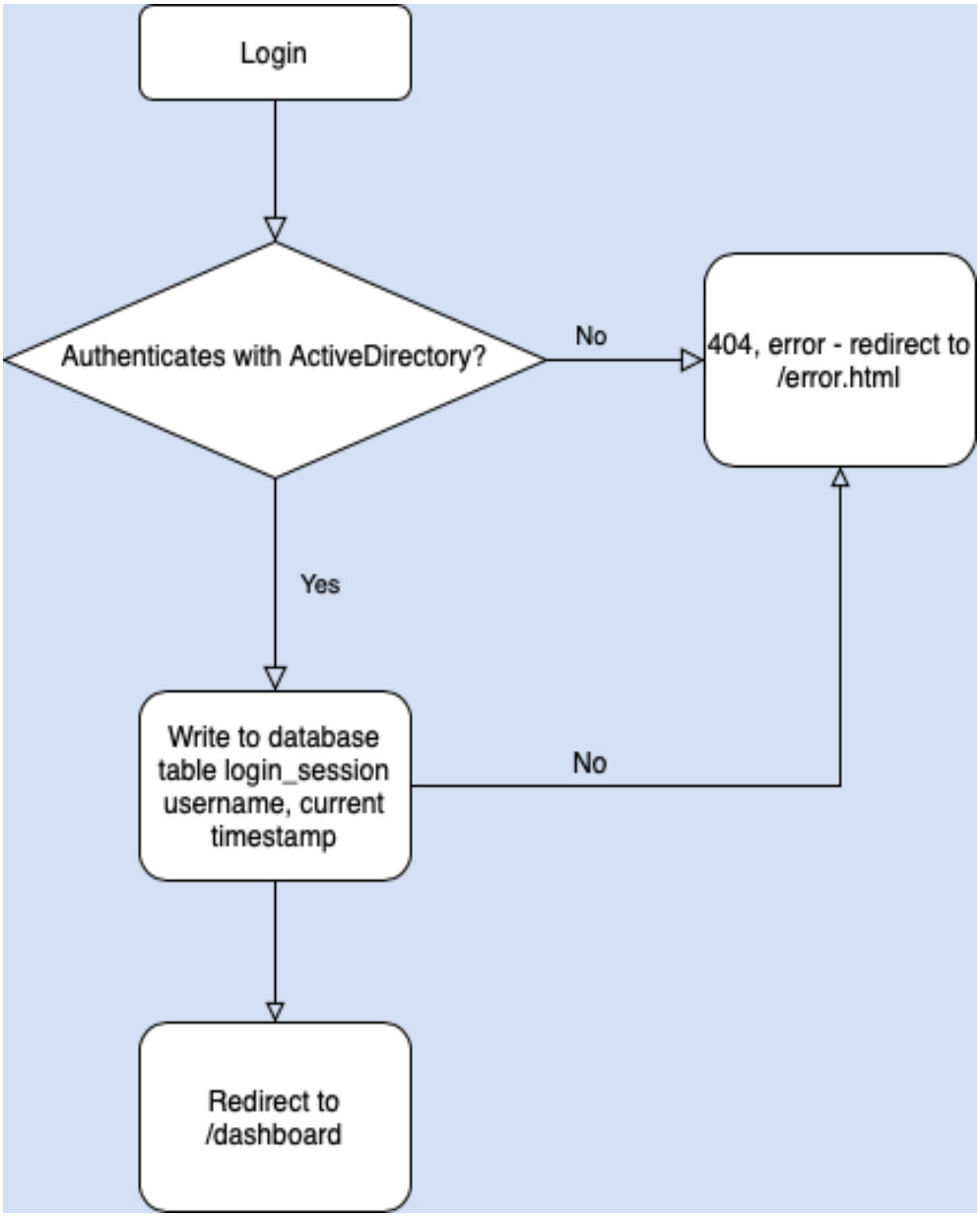

Figure 1: Sample Storyboard

D.2. NETWORK DIAGRAM

The requirement of being able to scale to 500 concurrent users is addressed by the below network diagram. Amazon Web Services will have a Burst Cloud on-demand which the Presentation segment will allocate from when a watermark of 80% concurrent users is reached.

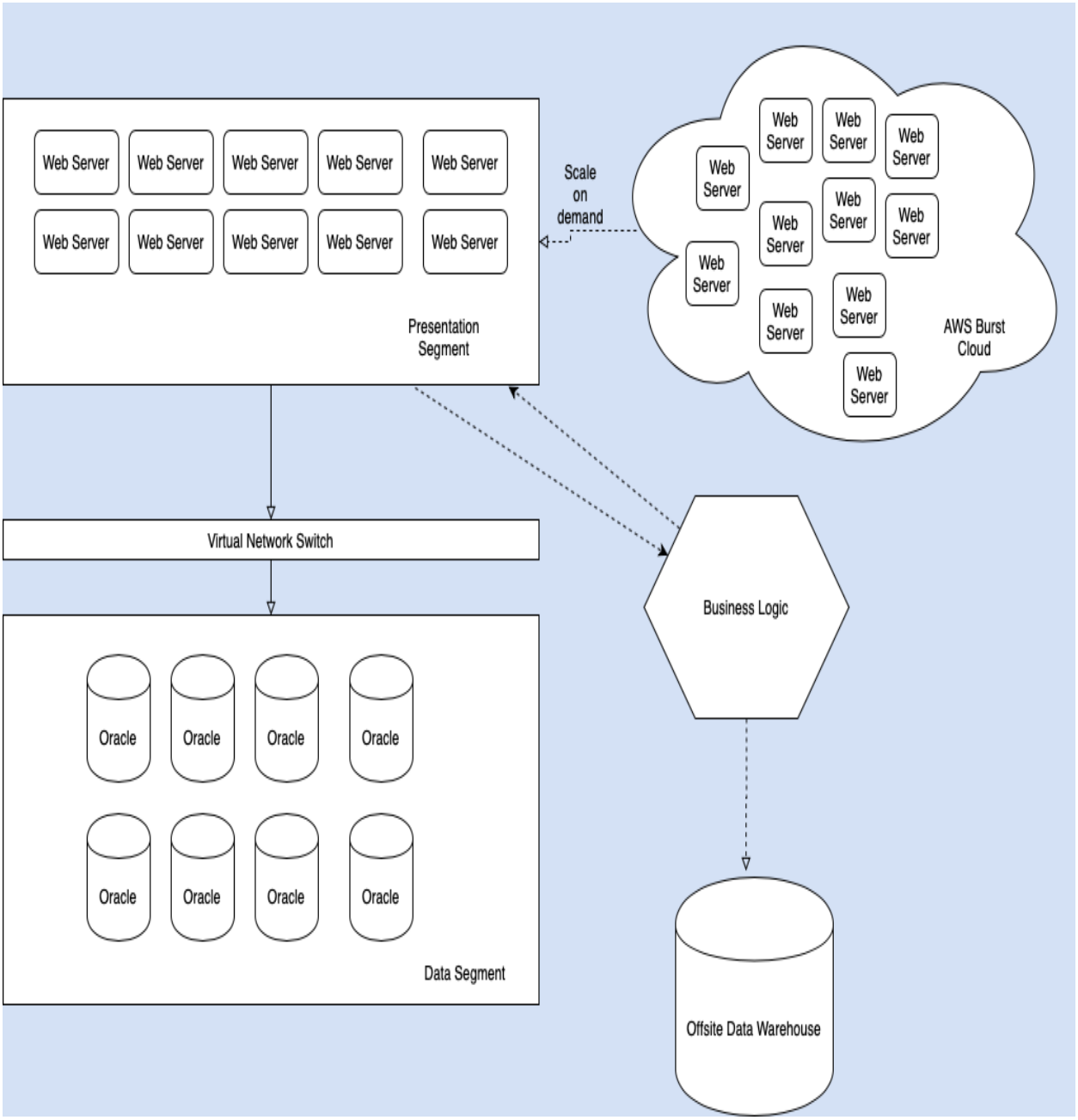

Figure 2: Sample Database

E. TESTING

Testing is necessary to ensure quality of software. In order to validate the ActiveDirectory requirement, Soft Delete requirement and keeping an audit trail of users logins the following functional tests will be conducted:

E.1. FUNCTIONAL TESTS

E.1.1. ACTIVEDIRECTORY LOGIN TEST

This will test a login with a username/password from the company's Active Directory
Preconditions: Conditions that must be present before test case can successfully run: Must have a enabled user/password combination in AVG's Active Directory.
Steps: The steps the tester must execute to test the feature. 1. Open a browser and browse to https://portal.avg-co.com 2. Click on username field and enter Active Directory username 3. Click on password field and enter password associated with username. 4. Click on Log Me In button under password field. 5. Take note of the redirected url.
The expected results are: - Page should be redirected to https://portal.avg-co.com/dashboard with a 302 return code.
The test should be marked as PASSED if the results are the following:. - Page should be redirected to https://portal.avg-co.com/dashboard with a 302 return code. The test should be marked as FAILED if the results are the following: - The return code is anything other than 302

A.1.1. SOFT DELETE TEST

This will test the ability to remove something from the view (Soft Delete).

Preconditions: Conditions that must be present before test case can successfully run:

User must be logged into https://portal.avg-co.com

Must have added at least one contact to Contacts section

Steps: The steps the tester must execute to test the feature.

1. Click on dashboard
2. Select Contacts from the left pane
3. Select a contact from list
4. Click on Remove from View button
5. Take screenshot

The expected results are:

- The contact will not be present in screenshot
- In the contacts table in the database there will be an entry for the contact that you clicked remove from view and it's viewable flag will be set to false.

The test should be marked as PASSED if the results are the following:

- The contact will not be present in screenshot
- In the contacts table in the database there will be an entry for the contact that you clicked remove from view and it's viewable flag will be set to false.

The test should be marked as FAILED if the results are the following:

- The contact you clicked Remove from View is still listed in the screenshot
- Upon inspection of the database table contacts the contact you specified is either non-existent or the viewable flag is set to true.

A.1.2. USER AUDIT LOG TEST

This is a test that a user who logs in is being tracked (User Audit Log)

Preconditions: Conditions that must be present before test case can successfully run:

- Must use Google Chrome, Internet Explorer or Safari browser.
- Valid username and password.
- Browse to https://portal.avg-co.com

Steps: The steps the tester must execute to test the feature.

1. Fill in text box next to username field with a valid username.
2. Fill in text box next to password field with a valid password.
3. Click on Log Me In button
4. Note URL

Expected results:

- Page should be redirected to https://portal.avg-co.com/dashboard with a 302 return code.
- The table named login_session in the database should be updated with username and current timestamp.

The test should be marked as PASSED if the results are the following:

- The table named login_session in the database should be updated with username and current timestamp.
- Page should be redirected to https://portal.avg-co.com/dashboard with a 302 return code.

The test should be marked as FAILED if the results are the following:

- The login_session table is not updated with username and current timestamp.
- The return code is anything other than 302

B. SOURCES

UCertify. Beginning Software Engineering. Section 12.3 Section Advantages and Disadvantages retrieved 06/21/2020, https://wgu.ucertify.com/?func=ebook&chapter_no=14#02TfW