Requirements Specification

12/6/2019

DigiTool Inc.

Version 2.0

Project Sponsor: Xi Zhou

Team Faculty Mentor: Fabio Santos

Team Members

Traybourne North
Noah Baxter
Spencer Clark
Langqing Dai
Miguel Quinones

Accepted as baseline requirements for the project:

For the Client:_____ For the Team:_____

Table Of Contents

1.0 Introduction

1.1 Project Background

The Digital Logic course here at NAU is one of the most important courses an engineering student can take, whether they're Mechanical Engineers, Electrical Engineers, etc. The course teaches students about several important logical reasoning concepts such as Karnaugh Maps, truth tables, and switching functions. Naturally, students might encounter issues when dealing with these subjects, such as trying to understand them or understanding why it is important to learn these subjects. Just this past semester alone, there were 75 students enrolled in the course, so we estimate that half of the students could struggle with the course topics every semester. Because of this, these students may have trouble participating in the course. Therefore, it is important that they are able to understand these topics immediately.

1.2 Project Sponsor

Our project sponsor is Dr. Xi Zhou, an Assistant Professor of Practice here at NAU. He teaches the Intro to Digital Logic course on campus and is very invested in the issues that students have with the course. He has explained that the course is structured so that the topics described above are introduced and worked through one at a time. In other words, students begin with Karnaugh Maps and truth tables and from there, they move onto switching functions and so on. For each topic, students practice working on examples and homework in order to better understand the topic at hand before moving on. This workflow follows the same general structure as most courses. We go into more detail regarding our sponsor's problems in the next section.

2.0 Problem Statement

2.1 Sponsor's Workflow

As stated in the previous section, the general workflow of our sponsor's course is similar to most other classes. However, because this is a standard course, there is a limited

amount of time allocated to it in the day. In other words, students only have the length of time allotted by the course to work through the topics. While the sponsor does host office hours, these might not always be suitable for every student. This leads us into the main problem, which is that students need another tool that could assist them outside of class time and office hours in furthering their understanding of the class topics.

2.2 Problems and Issues

The major problems our sponsor needs us to solve are the problems that students might be having with the course. Students having issues understanding the topics of the course is the biggest problem. This is due to the fact that the course introduces many new topics to students at a fast rate which makes it hard for students to learn and understand. The major problems are:

- The sudden change from the decimal system to the binary system causes a lot of confusion for students.
- The course topics are based on using binary, so if students have trouble adjusting to the binary system then they struggle with the entire course.
- One topic that causes consistent trouble for students is working with logical reasoning tools, such as Karnaugh maps, translating truth tables, and switching functions.
- Another issue was that a Java applet was created previously for students to use to practice the course topics, but Java applets are no longer functional in browsers so a new tool is needed.
- Similar tools to the applet exist online, but a large majority of them were either problem solvers or just digital textbooks. In other words, they didn't provide any real educational value to the students.

Due to the many problems stated above, our sponsor has given us the opportunity to create a web application that students can use to practice a majority of the topics discussed in the course and we will discuss a solution to these problems in the next section.

3.0 Solution Vision

We need to collect student's information (name, ID, usages.etc) and the information about each concept covered in the course (k-maps, truth tables, other digital logic material, etc). Our system will then output the questions linked to what was talked about in class and it will generate feedback that lets you know how well you did after you finish each topic in our web application. You will be given multiple attempts for each question and a user's accuracy for each answer correct will determine whether they move on from one topic to the next.

The main computational process of this system is to judge the correctness of the student's test, so it needs to compare the answer entered by the student with the correct answer that comes with the translation, thereby causing alignment (right or wrong). Of course, in the future, this valuable content will be enriched and some of it will point out the wrong answer.

Our client has more than 100 students, so he wanted a convenient way for students to have the opportunity to practice the course content. Our system will greatly reduce customer workload and improve efficiency. It reduces the customer's repetitive work when marking assignments and the system's unique way of answering will also help. Students only have to complete tasks in order of difficulty, so customers only need to check the last question of each student to know the information.

We have also considered other ways, such as developing a web page through PHP. This operation seems simple, but the updates within our web application may cause problems since our client doesn't specialize in programming. In the end, we decided to use JavaScript code. We will use a server that can be accessed by our client and we will also try to make it easier for our client to add / remove modules, but as of right now, our main priority is making it 100% functional and operational for a user only.

Our software allows students to:

- Learn / practice information through a developed interactive learning platform.
- Use a web application that contains multiple digital logic concepts all within one site.

Our software also allows professors to:

- Review the use of students and summarize teaching feedback to make improvements to subsequent teaching.
- Adjust the pacing of topics taught in the classroom depending on the student's understanding.

We believe that this software will reduce the difficulty of the Introduction to Digital Logic course, improve the pass rate of students, and reduce the work pressure from course instructors. For more information about our software's technologies and goals please refer to our technological feasibility document found on the NAU 2019-2020 computer science capstone page.

4.0 Project Requirements

With our solution vision planned out, functional, performance, and environmental requirements are needed to meet our client's needs and goals. Before thinking of our functional, performance, and environmental requirements, we first had to understand our domain level requirements within our system which include:

- User control and the ability to learn information
- The ability to check student's progress toward module completion as well as level progression within each module

These two domain level requirements will be broken down into detail in both the functional requirements and performance requirements.

4.1 Functional Requirements

Our client went over a couple requirements which consist of:

- The system will have a function that stores and validates user login information
- The system will have a function that determines whether a user has mastered a module or not before moving onto the next module
- The system will keep track of the number of attempts a student has used for each question
- The system will have a tracker which allows a student's progress to stay saved when logging in and out of our web application
- The system will have functions that allow students to practice module questions before attempting the real module
- The system will have a function that gives stars to the user based on the number of missed attempts they used for each question
- The system will have a function that saves and loads a user's work

4.11 User Registration / Login Screen

A user registration / login screen will be created to allow a user to access modules once their username and passwords match with those stored within our database. This will be done through a welcome page where there are two user input lines. These lines will either say "Username" or "Password" and if they have not created a user account, there will be a registration link underneath the word "Password". The link will direct the user to a page will they can successfully create a user account and afterwards they will be able to successfully sign in to our web application.

Once the username and password get saved into our database, we will use a hash algorithm to scramble each password created by a user. The reason why we decided to do this is because if a data breach were to happen to our database, the breacher would only have access to our usernames, but they wouldn't have access to our passwords. Each password would be scrambled into a long list of characters and numbers which would make it impossible to decrypt.

4.12 Module Mastery System

Each module covers one of the following concepts (more will be added if necessary):

- Karnaugh Maps
- Circuits
- Numeric Conversions (hexadecimal to decimal conversions, binary to hexadecimal conversions, etc.)

Each module consists of a leveling system that ranges anywhere from four to seven levels per module. The maximum number of stars a user can get on any level is three and amount of stars given to a user depends on how accurate they can answer the question. The more attempts it takes the user to answer a question, the less stars they will receive. For example, if a user were to answer the question right the first time, they would receive three stars and if they were to answer the questions right on their third attempt, they would receive one star. A mastery module score would be based off of the number levels provided in the module and the amount of stars a user got in a certain module. For example, if a module had five levels and a user could get three stars maximum per level, the highest mastery score could be fifteen out of fifteen stars. If our client made it to where a user has to receive thirteen out of fifteen stars to master a module, then we would have to make a function that would accommodate that. We would also have to make functions that tally up missed attempts which would involve verifying whether the user's answer is right not or not once they hit "Submit".

If they are correct, a function would be called which gives the user the correct amount of stars based off of their missed attempts. Once a user reaches the end of a module, another function will be called which would tally up the total stars a user has gotten and compare it with the overall mastery star score needed in order to complete the module. If the user's star total is equal to or greater than what is required they will complete the module. On the other hand, if they fail to meet the module mastery star count they will have to retake the module in order to move onto the next module.

4.13 Answer Attempts

A small overview was given in the subsection above, but this subsection goes into more detail as to how the answer attempt function will work. After talking with our client, the number of answer attempts in each level will be 3. If a user gets the answer correct, they will move onto the next question within the module, but if a user gets the answer wrong, a message will be displayed underneath the "Submit" button that lets them know that they got the answer wrong. They will also be able to answer the question several more times and if they run out of attempts, they will receive a low star store.

Since our client only wants to keep track of missed attempts rather than a timer, we would have to create a function that keeps track of missed attempts. For example, we'd set a three star mastery score to zero missed attempts, a two star mastery score to one missed attempt, and a one star mastery score to two missed attempts or three missed attempts. Then from there, a user would automatically move on to the next level or if they're already at the last level, they would receive their module mastery score.

4.14 User Tracker

A user tracker will also be implemented to save and load the game state of an individual user as they log in and out of our web application. This will be done through a function that saves the progress of a user once they click log out and from there, that saved game state will be stored either in a database or in a file. This can be done through a binary formatter which is used to serialize objects (0s and 1s) and save / load data in files. Once our team initializes a binary formatter, we would also have to create a path that files can be saved into. After a file path has been created, we would then have to create a FileStream which enables data to be contained within a file. From there, the given FileStream would have to be connected to the created file path and then user data can be saved onto the created file. Once the user data has been added to the given file, the data inside the file would have to be converted into binary with the binary formatter that was originally initialized.

Another function would also have to be created that can load data from a user once they log in. A user has to already have started a module in order before data to be loaded and if they haven't started a module yet, then they would just be presented with the homepage. On the other hand, if a user were to have already started a module, their data would be loaded and they could resume where they originally left off. This would be done by initializing a binary formatter and creating a FileStream which allows data from a file to be opened. Once the data from a file has been opened, the initialized binary formatter can be used to deserialize the binary data stored within the file. After the binary data has been deserialized, the user game state can be loaded on the screen. The user will then be able to resume where they left off.

4.15 Practice Mode Option

The last feature that a user should have is the ability to practice material from a module before actually starting a module. This can be extremely useful if a user were to continuously fail a module over and over again. A function would have to be created which would give the user a list of modules to choose from. Once the user has selected a module, they will then be provided with a random question that is similar to an actual question given in the actual module. Similar buttons will be provided ("Submit", "Reset", "Hint") and once they answer the question, a simulation star rating will be provided or they will see an incorrect answer message.

The way we design this practice mode option will be very similar to how we design an actual module. The main difference is that the user won't advance in module levels. Instead they will just get one random question from a random level within the module. If they answer correctly, the user will be returned to the practice mode module menu, where they can choose another practice problem from any of the provided modules. The practice mode tab will be integrated in the horizontal navigation bar provided at the top of the screen so it can clearly be available once the user logs into his / her account.

4.2 Non-Functional Requirements

Non-Functional requirements that should be noted while developing this web application are:

- Web application must be secure and safe to use
- A user interface which allows users to interact which the web application
- System must support 150 users at the same time without affecting the web application's performance in a large way
- Response time must be between 0.1 - 2.5 seconds
- System must be operational from August - December & January - July

4.21 Secure Website

A secure website is something important to test as our web application is being developed. As mentioned in section 3.1, the only important information that could be accessed is a user's password, but we already discussed how user passwords can remain secure in terms of the web application. The best thing to protect our web application would be to use a web application firewall which could filter and block unwanted HTTP traffic. A web application firewall can also protect users against SQL injections, cross-site scripting attacks, and more.

Another method that our team could look into would be the configuration of security. Our team has to make sure that application servers, web servers, database servers, and the platform itself have a secure configuration properly implemented so that attacks can occur less. We do understand that we would be using an NAU server to possibly host our website since each faculty partner has enough storage in which web applications can be stored so that shouldn't be a problem. On the other hand, our team does have to think of alternatives in case our client can't host our web application. We have been thinking of using an online server to host our web application, but we have to do a bit more research before making that decision.

4.22 User Interface

A user interface will also be implemented to allow a user to easily navigate between tabs that are available in our web application. In terms of an interface, we want to make a graphical user interface that is designed for a computer / desktop because that is what our client wants. Once the user logs into our web application, they will be provided with a homepage which contains details about our web application and what it has to offer. The homepage will also contain a home tab, modules tab, practice mode tab, and a log out tab. These tabs allow a user to either return to the homepage, access / practice material from a module, or sign out of their account. The tabs will be designed in a horizontal navigation bar located at the top left of the screen. As the user hovers over each tab, it will slightly change color to indicate that the user's mouse is currently on a specific tab. As we progress further into our web application, more tabs may be added, but as of right now, that is all that is needed.

Within each module there will be buttons a user can click which are labeled, "Submit", "Reset", and "Hint". The "Submit" button will be used to submit a current answer that a user has entered, while the "Reset" button will reset any answer that the user has entered. Some module problems may need multiple answers to be inputted into answer boxes. The "Reset" button makes it faster to delete all answers since the user won't have to click into every answer box and manually delete the answer themselves. Lastly, the "Hint" button gives the user a clue as to how to solve the problem on a given level.

4.23 Web Application User Capacity

Our web application should also be able to hold 150 users at one time without affecting the web application's performance. The number 150 was generated based off of the statistics from NAU pairs. From the year the original capstone project was created (2016) to now (2019) our team looked at the number of students that were enrolled in the Introduction to Digital Logic course from 2016 to now. The largest amount of students enrolled in the Introduction to Digital Logic for that given time period was approximately 120 students. Once our team was able to find that out,we added a couple more users to that number, which was an additional 30 students, to simulate a small increase in students that could potentially happen in the future. We wouldn't be able to test user capacity until

our web application has been fully implemented. the best thing to do would be to have our client test our website in his lecture and give us feedback. Since our web application plans to be hosted on the NAU server, there shouldn't be any major issues with the amount of users that could use a web application at one time. Its performance shouldn't be affected at all, unless a user doesn't have strong network connection.

4.24 Response Time

Response time must also be between 0.1 and 2.5 seconds. The amount of users that are on our website at the same time and the strength of a user's network connectivity also plays a part with the response time hitting our given range. A way we could increase response time is to optimize our database by ensuring that our queries are written efficiently and by storing what only needs to be stored. Having unnecessary things stored in our database could lengthen the time in which a response is carried out. Another thing that we can do to ensure fast response times is to check our host and web server to make sure that it's fast and sufficient.

4.25 System Operation

Our web application must be operational from August through December and from January through July. This is because these months are the times in which the course that correlates with this web application is taught. In order to ensure that our system operates within these months, we would have to look at servers and the percentage in which they're available for the user to access. We also have to look at the rate in which servers are maintained and would have to make sure that the times that they're being maintained don't interfere with the months that the system should be operational. Our main priority would be looking into the NAU server and the months in which they operate and the months in which maintenance may occur. From our team's previous experiences, maintenance rarely occurs. We also do understand that when maintenance does occur, maintenance only lasts for a couple days. An example can be when LOUIE was being repaired. It took about three days for Louie to be repaired and after that, it was able to be used for the rest of the year.

4.3 Environmental Requirements

When it comes to environmental requirements, there are a couple factors to consider which consist of:

- Modern Browser Support
- Device Compatibility
- Internet Connection

4.31 Modern Browser Support

Since our web application does use HTML5, modern browser support is heavily required. All browsers support HTML5 to some extent, but not all browsers support it completely. Below is a list of browsers that fully support HTML5.

- Internet Explorer 6 & above
- FireFox 3.5 & above
- Opera 11 & above
- Safari 5 & above
- Google Chrome 8 & above

Any computer or desktop that has the ability to install a modern web browser that fully supports HTML5 should have no problem with using our web application.

4.32 Device Compatibility

As of right now, our initial project design plans to only work for computers specifically. Our team is making the assumption that a majority of the students who are in the Intro to Digital Logic course have a computer on them or have access to computer since computers are provided for free on NAU's campus. Since we don't have a lot of time to make our web application compatible for both computers and mobile devices, our client told us to focus on making a fully operational web application for computers. If we do have enough time to make it portable towards mobile devices as well, we will do so, but as of right now, our primary goal is to make our web application operational towards

computers only. As long as a computer has between 100 - 500MB of memory needed to download a current web browser, it will work with our web application. Device specifications all depend on the browser that the user uses. If a user were to use Google Chrome as a browser, they would need to have the following specifications:

- Linux
    - 64-bit Ubuntu 14.04+, Debian 8+, openSUSE 13.3+, or Fedora Linux 24+
    - An Intel Pentium 4 processor or later that's SSE2 capable
- Mac
    - OS X Yosemite 10.10 or later
- Windows
    - Windows 7, Windows 8, Windows 8.1, Windows 10 or later
    - An Intel Pentium 4 processor or later that's SSE2 capable

Different specifications come with every browser, so if a user were to use a web browser other than Google Chrome, they would have to see if their computer specs meet the browser requirements.

4.33 Internet Connection

Internet connection is another important environmental requirement that is needed in order for a computer to run our web application. Since our web application will be connected to a server and an online database, no device will be able to use our web application successfully without the connection of the Internet. A user's Internet connection will also affect the speed in which web pages will be loaded and the response time it takes for an event to occur once a user presses a button.

5.0 Potential Risks

Overview

	Likelihood	Severity
Loss of information	Low-Medium	Medium-High
Database availability	Low	Medium
System incompatibilities	Low	Low-Medium
Inadequate UI	Medium	High
Rival Product	Low-Medium	High

Figure 1: List of risks and their likelihood/severity

5.1 Loss of personal information

Since we're working on a web application that needs to store user data, it is our ethical responsibility to protect the information our users give us. This data is mostly comprised of usage statistics, but may also contain emails and passwords. The usage statistics are unimportant as they give little information about the user or their personal life. Emails and passwords are much more important as they give some form of access to another person's data, especially since passwords are commonly reused.

We plan to protect this information with simple encoding features such as SSL or using google's login and have them protect the information for us. Losing the usage statistics will negatively impact our public reputation and emails and passwords will do the same as well as possible legal repercussions if we are seen as negligent.

5.2 Database availability

Since the web application will rely on using the database for saving progress, we need to make sure it remains usable as much as possible. Inability to use the database will result in users not able to retain progress made in modules. At best it is a minor inconvenience that gets fixed promptly and rarely happens. At worst, it becomes a recurring problem and significantly impacts the users ability to use the web application.

This risk is not concerning because database servers are reliable and we only need it for 2, 4 month periods per year. All other times during the year can be used for maintenance and updates.

5.3 System Incompatibilities

We can only ensure so much in the way of subsystems performing together. While this is unlikely, it will remain a threat until the project is done. In the event that some subsystems do not work together as planned, we may have to add new systems for features needed, or replace a system to prevent the incompatibilities. Finding an incompatibility would halt progress on the web application. Depending on the severity of the incompatibility, it could delay us for a short time of a day or a very long time of a month or more. The deciding factor is what breaks and how important it is to us.

Since we do not have many subsystems, it is unlikely that we will have a compatibility issue and that will delay us for long. One way we can mitigate external system incompatibilities is to have system/browser specifications to inform a user of a potential problem.

5.4 Inadequate UI

Our product is going to be “in contact” with our user for a considerable amount of time during the course. It is very important that the user can use and manage the web application effectively. The goal of the web application is directly related to its efficiency in helping people learn about digital logic. A bad UI can slow down the user, make them frustrated or confused, and overall dislike the program. The more the user dislikes the program, the less it helps them learn.

We can solve this by keeping in close contact with our client and even more towards our end users. Having feedback from end users is the only way to ensure that the UI is not a detriment to the web application. Since our client is a teacher and works with our end users, he can obtain testers to get suggestions from.

5.5 Rival product

Currently, our project is a specific solution. There is nothing else that solves our problem. The closest things are automatic solvers, which do not help the student learn and learning management systems, which none specialize in digital logic or focus on the practice element of learning. Grading is the only form of management.

In the event that a learning management system or some other large system decided it wanted to create a specialized tool for teaching and practicing digital logic, our current solution would need to be either perfect or changed and made more expandable. Such an event is potentially catastrophic to this project. Our best defense is to make sure our code stays flexible, while keeping track of potential competitors so we do not get blindsided.

6.0 Project Plan

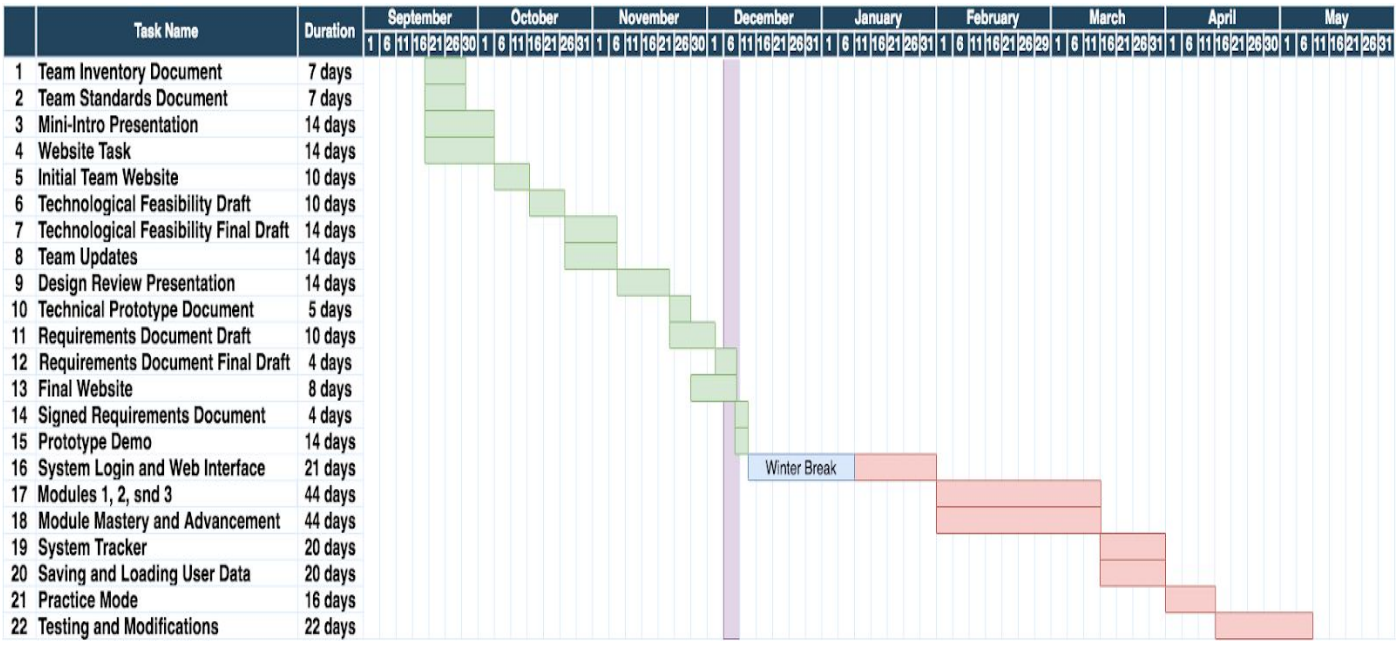

Figure 2: Our two semester project schedule

As of right now, we are currently working on our prototype to show to our client which will consist of handling usernames and passwords within our database and the functionality of module 1. We want to make sure that all buttons work and the user is properly directed to where they need to go if they were to click on a module or tab within our web application. After winter break, we plan to have 5 to 10 milestones which include completing all the modules that we need to complete, properly handling a user tracking system, and full functionality of our web application as a whole. Our main goal is to finish these milestones by April and then focus on testing /debugging our web application for the remainder of the semester.

7.0 Conclusion

To conclude, many students are still struggling with concepts discussed in the introduction to digital logic course taught at NAU. Although students have access to tools that can help them with learning, these tools only give out answers rather than having users practice it on their own.

To solve this problem, we want to create a web application that will not only help students with one concept, but with multiple concepts, so students can practice and learn material all from one tool. This web application will greatly benefit students since everything within our module will focus on topics discussed in class which will eliminate unnecessary material that could be taught to students if they were to use an online tool. Understanding and coming up with ways to solve functionality requirements, performance requirements, and environmental constraints really contributed towards our project's progress because we now have an idea of how to go about our project implementation.

From here on out, we could progress forward in our project prototype and after we receive feedback, we could further our development towards our project's overall goal, which is to create a self-study toolkit for students who are enrolled in the introduction to digital logic course. Although we have minor problems that will be fixed in the months to come, we are positive that we can meet the goals and requirements given to us by our client and we foresee a fully functional web application in the future.