CSCI 3010 - Project Proposal

Authors:
Alex Palo
Sam Goulding

1. Project Description

The main goal of this project is to create an application that helps individuals who are new to cyber security or individuals who want a better understanding of common cyber security topics. The app will be split up into multiple sections, each corresponding to a different topic that is related to cybersecurity. We will have sections covering Asymmetric Encryption (activity will include a Key Exchange exercise, information on RSA, and more) and Authentication (activity will include a password complexity exercise, information on Salting, and more). With enough time, we'd also like to implement sections that cover other areas of cyber security - ranging from a broad overview and applications of cyber security to more specific topics such as Social Engineering and Role Based Authentication.

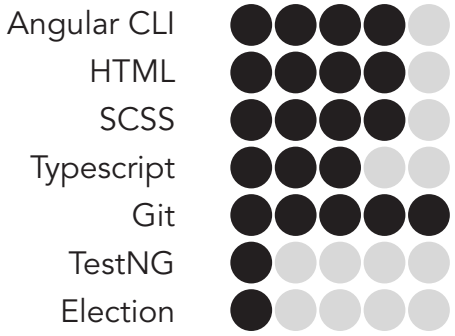
From a home/landing page, users will be able to navigate to any section of the app, where they will arrive at the section home page. From a section home page, users will be able to select from a number of interactive activities to help them learn about the specific section topic. The interactive activities could include short learning activities to test a user's knowledge, puzzles that require understanding of the topic, or interactive
examples of how the topic works (e.g. for cryptography, users could encrypt and decrypt messages using certain methods).

The project will be programmed using the Angular framework which incorporates Typescript, HTML, and SCSS. The logic for the application will be written in Typescript and the GUI will be designed with HTML and SCSS. Angular is intended to be used for creating web applications, so upon completion of the application, the app will either be hosted online via Github, or packaged into an executable desktop application with the use of Electron.

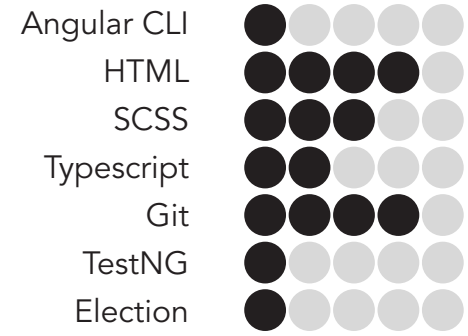
Throughout the process of programming the application, Github will be used for version control. Each page of the application will be created in a branch and then merged with master upon completion. If any bugs arise, the bug will be fixed in a dedicated branch for that bug and then merged with master. As a testing framework, TestNG, a framework specifically for testing applications with Angular, will be used. This project will use the Flyweight and Prototype design patterns.

2. Technologies

Alex:



Sam:



3. Essentials

The two most essential parts of our project are user interaction and the topics/material that we provide to the user. The whole idea of our project revolves around the user being able to engage with our application, helping them learn in various ways. Our project won't work if we don't have some form of user interaction, that is interaction beyond just opening and reading a wall of text. The user having the ability to switch topics, go back and forth between pages, and show they understand the material is of the utmost importance. If we aren't able to implement these general ideas, then we have to come up with another way to help the user learn.

The second important piece of our project concerns the material that we are including in our application. Even if we are able to implement the user interaction piece, if the material we provide the user is irrelevant, out of date, or something else, then the application isn't accomplishing what we want it to do. Ensuring the material is useful shouldn't be too hard, but if we can't find any material to provide to the user then we don't really have a project...

4. Resources

The outside resources that we require to complete our application include Cybersecurity learning material, and maybe some visuals/graphics to help the application convey the material in a useful manner.

We will need to gather accurate and relevant material that would be covered in your typical 'Introduction to Cybersecurity' class or, in general, material that we would consider helpful to someone unfamiliar with the cybersecurity space of Computer Science. Seeing that both Alex and I are in an introductory cybersecurity course, picking relevant topics to include in the application shouldn't be too much of a challenge. When looking for relevant topics to include, we can look to the material provided to us in that course, or we can look at other sources for relevant topics.

Incorporating visuals and graphics into our application would significantly improve the effectiveness of our application's ability to teach the material. Having said this, the visuals/graphics don't need to be masterpieces of art created by Pablo Picasso. Rather, the visuals/graphics should merely be there as a way to help guide the user through the various exercises we provide them. These visuals will probably end up being very rudimentary, simple illustrations like arrows or moving shapes that are included in exercises. Because of this, we can probably create the visuals ourselves, but if we need more detailed visuals to help guide an exercise, we have the internet at our fingertips.

Seeing we are not doing any data analysis or manipulation, we won't need to find a source for any data because we aren't using any!

5. Architecture

We are yet to make a final decision on how we are going to handle individual user interactions - will we incorporate accounts so users can track their progress? - but our idea right now is that we will provide a static information base to provide to every user. Due to this, we won't need to integrate any back end application, however it still may be useful if we choose to do so.

Our front end and user interface are synonymous seeing that we aren't implementing a database/back end functionality. Front end functionality will drive the entirety of user interaction, allowing the user to go to various sections/pages/activities, all of which will follow a basic template/structure according to the object models.
When we start implementing our front end, it will be imperative that we have a well-defined structure of how our object models interact with one another. Our general schema of object models and how they interact with one another is as follows:

Current Session Object:

- This object will be initiated each time the application is opened, where the intention is that user progress won't be saved if application is closed.
    - Saving user progress is to be looked into.
- Overall goal of this object is to track details about user progression through the different sections.
    - Contain fields for storing activities that have been completed.
    - Contain fields for storing activities that have not been completed.
- Store what activity the user is currently on.
- Store minor details about session.
    - User's name.
    - Time spent in session.
    - Etc.

Section Object:

- This object serves as a placeholder for any of the information/topics on
- Cybersecurity that we decide to include in the application.
- Object is comprised of a document object model that will be implemented in angular.
    - DOM includes:
        - Topic name.
        - General information about the topic.
        - Links to relevant external sources.
        - Internal link to Activity for the section.

Activity Object:

- Each instance of this object will be tied directly to an instance of a Section Object.
- Include a list of Page Objects.
    - This object will be able to iterate through the list of pages.
    - Current element of list will be the page displayed to user.
    - Each page object has a 'completed' tag, use these tags to update progress on activity.
- Include an iterator able to iterate forward/backward through list according to user interaction.
- Progress attribute that shows the user's progression through the activity.

Page Object:

- Similar to the Section Object, this object will be comprised of a document object model and a progress flag.
    - DOM includes:
        - Text
        - Navigation left button
        - Navigation right button
- Progress flag.
    - If the user navigates to the page, then away from the page, set progress flag to 'completed.'

We intend on incorporating the Flyweight and Prototype design patterns when we start to build our application. Both of these patterns seem rather intuitive to incorporate into any project design, but are especially applicable in our case. We'll do our best to explain these design patterns; Factory design pattern is the idea that we can use a static member-function to create and return instances of a class, hiding the details of the class from the user. We will be able to incorporate this concept when we look to create multiple instances of a page, for example. We will be able to simply call the function and it will create an instance of that object, in our case a default topic home page maybe. Prototype design pattern is where we copy an existing object rather than create a new one from scratch, avoiding operations that are costly. We will be able to incorporate this concept in tandem with the Factory design pattern. Rather than creating a new page object every time we need another page, we can simply copy the last page we used and avoid having to initialize things like the progress bar, text boxes, or images.

6. Interface Design

The GUI will rely on a landing page that the user will be directed to when first starting the application. From the landing page, there will be buttons that can be selected to navigate to any of the sections within the application. The buttons will contain a brief description of the section, the section title, and an image that is relevant to that section. Clicking one of these buttons will direct the user to a specific section page. All section pages will follow the same layout with a large tile to direct the user to that section's "learn" page and then several tiles to direct the user to an interactive activity.

See examples on next page...

Home

Section Title

Brief description of section and a photo.

Section Title

Brief description of section and a photo.

Section Title

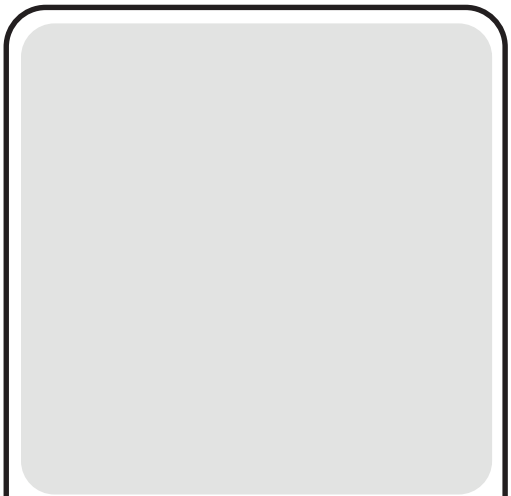
Brief description of section and a photo.

Section Title

Brief description of section and a photo.

SECTION

LEARN



Description about what will be taught in the short course.

ACTIVITY

Description about the contents of the activity.

ACTIVITY

Description about the contents of the activity.

ACTIVITY

Description about the contents of the activity.

7. Detailed Plan

Homework 2, Part 3 February 23rd:

By this due date, we hope to have finalized the end goal of the project. This will include what the application looks like, how a user interacts with it, and what material we will be covering. We hope to have solidified our understanding of how the application will work, allowing us to focus on the implementation piece of the project.

At this point, the user won't be able to interact with/see/be able to do anything other than look at the source code of the project (assuming we have written some by this point)

In order to accomplish this goal, we will need to discuss what material we think is going to be relevant for the application. This means we will need to generate at least 4-5 topics to include on the application and do some research into these topics so we can start to get an idea of how we convey the material to the user.

Deciding how the application will look and how the user interacts with it will also need to be discussed, but won't take as much research (if at all) in order for us to move forward.

For most of the material to be included on the site, we will be able to gather enough information from our own understanding of the material, coupling this with external sources should render the material solid.

Turn in: By this date, we will have set up our public Github repo appropriately and have it shared with the instructors. We will turn in our revised proposal on Canvas, and also have it in our checkpoints folder on Github.

Week of March 1st:

By this due date, we hope to have the foundation for the application completed.

This means we have setup the general layout in Angular - laying out the different areas we can implement in the future.

The user won't be able to interact with anything yet

In order to accomplish this, we will need to have decided on how the website will function and have a basic idea of how Angular works.

Alex is very familiar with Angular, but I (Sam) will need to familiarize myself with how the framework works.

Week of March 8th:

We will hope to have created a basic template for how each section is going to be organized.

The user might be able to open up a page of how a section is going to work.

In order to accomplish this, we will be implementing the Angular framework using Typescript for our integration layer.

As stated previously, Angular is familiar to us at this point, but we will need to learn how Typescript works.

Homework 3 - March 13th:

By this due date, we hope to have the general section template completed, and will have the ability to apply that template to whatever topics we wish to add into the application.

The user will be able to interact with a generic template of a section.

In order to accomplish this, our Angular framework will need to have the HTML/SCSS/Typescript portions of the template completed.

Turn in: For the turn in on this date, we will be uploading the front end code that is used to generate this general section template.

Week of March 22nd:

By this due date, we hope to have inserted a template for a topic into the over schema of the website. This will include being able to navigate from the main page to a specific topic, and walking through the activities of that topic.

The user will be able to go from the main page to a topic and navigate the topic

The requirements for us to accomplish this are similar to the previous milestones - implementing our front end side of the application

Week of March 29th:

By this due date, we hope to have the first version of application completed. This will include various topics implemented into the application, ability to navigate through the application, and serve as a marker for what we need to change about the website.

The user will be able to navigate the application, going to various topics and using the various features included in the application.

The requirements for us to accomplish this are similar to the previous milestones, however we will need to be diligent in taking not of bugs/issues that crop up during this phase.

Homework 4 - April 3rd:

By this due date, we hope to have come up with a list of things we can change about the application.

The user will still only have access to what we have previously accomplished.

In order for us to accomplish, we will need to test our application for flaws/bugs that come up during general navigation of the application.

We will need to learn how we can implement unit testing into the Angular framework so that we can do a good job testing the application out.

Turn in: For the turn in on this date, we will have pushed the current version of the application to our repo (which you have access to) along with a document highlighting things we would like to change in the application.

Week of April 12th:

By this due date, we hope to have gotten a start on the improvements for the application. Specifically, having approximately half of the issues ironed out by this date would be optimal. On top of this, we will want to have started user testing to help us understand what bugs/issues we have.

The user will be able to interact with the application like beforehand, but will hopefully have a more enjoyable time due to the changes made to the application.

In order to accomplish this, we will need to have utilized our testing method for the application. From this, we will need to be able to understand unit testing, or testing in general, with Angular.

Homework 5, checkpoint - April 17th:

By this due date, we hope to have covered all of our known issues we highlighted a couple weeks back. The major bugs/issues will have been squashed, and the application should be running smoothly.

The user should have a near seamless experience when using the application. Most, if not all of the features should be error free, and the application as a whole should be coming together.

Similar requirements to accomplish this as last week - need to continue testing with Angular.

Turn in: For the turn in on this date, we will have pushed the current version of the application to our repo, with (hopefully) all of the known issues fixed. If we still have issues to fix, we will also submit a document highlighting all the issues that we have fixed and issues that are yet to be fixed.

Homework 5, final due date - April 29th:

At this point, I'm not too sure what we will need to have done. There is a lot of variability here, not just in where we will find ourselves, but what we will need to have completed. In my opinion, this date serves as a major milestone for us and will allow us to take a step back and understand where we currently stand and what else we need to accomplish.

Turn in: By the final due date, we will have pushed the most recent version of the application to our Github repo. As far as submission goes, we will submit a final document that highlights what we have accomplished, what still needs to be accomplished (and how we will do it), and other details/digressions about the project as a whole.

8. Class Engagement

Both Alex and Sam intend on attending lecture at the same rate that they were previously attending at. The programming exercises are directly related to the material covered in lecture, so we'd be stupid to blow off lecture because then not only would we miss participation points, we'd also be putting ourselves at a disadvantage when looking to complete the programming exercies.