Progress Report

- Increment 3 -
Group #10

1) Team Members

Name	*FSUID*	*Github ID*
Aidan Martin	*am16br@my.fsu.edu*	*am16br*
Darren Kopacz	*dk16d@my.fsu.edu*	*dk16d*
Riley Corey	*rjc18d@my.fsu.edu*	*rileyycoreyy*
Douglas Kendall	*djk12@my.fsu.edu*	*dougkendall*
James Kerrigan	*jk18ed@my.fsu.edu*	*KerriganJames89*

2) Project Title and Description

Our application, WebSight, is a web based software that allows users to monitor a list of websites. Our platform allows users to: create an account, add website URLs to a list, receive updates any time a site is updated, manage alerts, and change settings. These settings and preferences include preferred method of notifications, timeframe to check for updates/receive alerts, and changing tracking-expiration date for sites.

3) Accomplishments and overall project status during this increment

In this increment, we managed to get the localDB working with the help of Entity Framework, as well as finish up some of the logic that's needed to communicate from client to server. It's a completely new project built up from our previous two as compatibility issues were a problem across operating systems. It's created from an empty project, unlike our previous templates where much of the back-end logic was already established. However, our previous template was used to configure functionality for our site but was not integrated with the build completed by James as we wished. The site allows users to manually check for website updates!

4) Challenges, changes in the plan and scope of the project and things that went wrong during this increment

From our previous increment:

We struggled with collaboration and scheduling meetings due to the remote nature of this class with group members not all present on FSU's campus. We also ran into issues running the ASP.NET on

Mac devices, since the Mac version of Visual Studio does not support the current version of ASP.NET. We worked to transfer the application to the universally accepted ASP.NET Core, but have had some issues with recreating the application in the new environment.

The Visual Studio application available for Macs does not offer the same tools and interface. It does not offer the Server Explorer used to create our databases and tables. It also does not let you see the design view of our application due to the fact that the designer is also not supported on the Mac version. So the Mac users on our team were very limited on what they could help with. However, many had other PCs they could use to contribute more after hours of trying to find the Mac ways of doing certain tasks.

This transition has broken some functionality, such as our login page, which will require additional work reconstructing it. We are currently debating ways to merge our previous framework without much loss.

Robust testing on SendGrid is unavailable at the moment, but we worked on sending emails using Sendgrid to better understand future integration

new:

We faced many challenges in this increment, mostly stemming from the use of C# and the use of the ASP.NET framework, which is not functional on Mac machines. This issue, coupled with virtual learning and lack of access to Windows machines, caused many problems in our group. Without being able to meet or make use of FSU facilities and equipment, we were unable to complete all of the goals we set out to complete. Scrambling to produce a functional piece of software, we decided to discard many of the extra features we set out to implement and spent more time focusing on the core of the application.

This includes continuing to host locally rather than on Azure. An Azure SQL Server and Database was created, but was deprecated and resulted in WebSight not being able to connect with it.

From our current/last increment:

We decided to revert once again back to our previous framework build since we lost a lot of functionality. However, this created much of a challenge as the final stretch of the semester was near and building up the project again was harder than we thought. The conversion was a failure, we decided to ditch the template and start anew, using what was compatible with from our previous framework. This empty project was built up from a lot of code provided by Professor Mill's CIS4930 course. It uses multiple logical layers; an Enterprise controller with the help of Entity Framework. This was still very hard to accomplish with how much time was left in the semester, but this was overall a better idea than using Microsoft's Identity; a template that we were all too inexperienced to use.

Functionality was also added to our previous framework build aside from Jame's build. We added functionality to our URL bar which allows a URL to be entered. I configured our SQL database with tables to hold the information. When the URL is inputted, a HTML scrape of the website is made and the HTML string is outputted as a pop-up box. The HTML scrape is then hashed into a 128-bit fingerprint using the MD5 hash and the hash is then outputted to the screen as well. If the hash is *recognized in our data table for that URL, the URL has not been updated and a message appears to the user. If the scrape is different from the one stored it has been updated and the new hash is now saved. I also made a GUI for our manage alerts page which was to be integrated with our notifications API. We were not able to configure the API or the login aspects.*

5) Team Member Contribution for this increment

a) *Progress report,* *including the sections they wrote or contributed to*

James: *Accomplishments:* *Explained the new framework.* Challenges, changes in the plan and scope of the project and things that went wrong during this increment: Added my challenges, and my overall opinion on the difficulties of this project.

Aidan: *Project Title and Description: Described the nature of our project. Challenges, changes in plan and scope of the project and things that went wrong: Added challenges we faced during this increment stemming from machine limitations and distance learning issues.*

Darren: *Project Title and Description: Grammatical corrections to project description . Challenges, changes in plan and scope of the project and things that went wrong: Added info on Azure hosting.*

Riley: *Accomplishments: Explained the addition to our previous framework and the added functionality. Challenges, changes in plan and scope of the project and things that went wrong: Explained what was changed to the previous framework for this increment and what we were not able to do.*

b) *Requirements and Design Document (RD)*

(Consists of information pertaining to the previous increment; no changes have been made to these initial documentations)

Riley*: Use Case Diagram: I diagramed our Use Case Diagram in order to organize our system/software requirements for our web application. It was important to create this in order to visualize and organize the implementation parts we need especially when our software is not in development yet. It is also helpful to see how our cases interact with one other in order to begin the integration of their relationships in our development.*

Darren*: Class Diagram: Since C# is a heavily object-oriented language, I created a simple class diagram illustrating the main entities with their properties and methods that will be used in the project. At the moment, there is currently Account, Site, and Manager. Manager will be used to keep track of users and perform administrator levels tasks. More classes will likely need to be added to this diagram in the next increment as we begin working with the web-alert system.*

Functional Requirements: I added a list of functional requirements broken into three main categories: Login Page, Main Page, and Make Changes Page. Each page has the requirements listed in order of high to low priority.

Non-Functional Requirements: Non-functionals were broken into three categories: Security, Safety, and Performance. Since users expect to be able to save and modify their list of tracked websites at any time, WebSight needs to be secure enough to ensure users can only access their own accounts. This also prevents malicious or accidental misuse of the alert feature to spam other people's emails or phone.

Operating Environment; Assumptions & Dependencies: Added info on the current operating environment and planned API dependencies.

***James:** ER Diagram: I created a basic three table diagram: Users, Accounts, Changelog. The graph represents a very minimal presentation of how data will be stored in our tables and the relationship between each table. Users will have to create a unique username, and a password. Each user will create their own unique account; a specific ID, and perhaps more personal information can be appended to this table. The Changelog (a master list) will be the archive for URLs, which can only be accessed by the accounts that submitted them. From here, it needs to be decided if compared data should be reinserted back into a table, else all data comparisons will only be done when a user enables the specific query.*

***Aidan:** Contributed to conversations regarding non-functional and functional requirements. Overview: Added a synopsis of our project, including its functionality and set-up. Class Diagram: added textual description to clarify diagram and links between classes. Operating Environment; edited to support our current version.*

***Doug:** Comments for Use Case diagram*

c) Implementation and Testing Document (IT)

***James:** I've commented in every single section about the newly created framework and the difficulties I had re-implementing some of our previous features.*

***Riley:** Platforms, APIs, Databases, and other technologies used: I commented on what technology I used to configure the functionality of our web application.*

***Aidan:** Platforms, APIs, Databases, and other Technologies Used: Added to support the current version. Execution-based Non-Functional Testing: Completed section to reflect testing completed and current state of the project.*

d) Source Code

***James:** Created an entirely new Framework for the final increment. Added client side code, communication controller, an Enterprise Controller, and server-side logic. I've also updated the HTML and included some Javascript/Angular to display the code into our tables when the server is launched. My experience with this stuff is relatively new as this is the first semester; I've only* *used PHP/HTML in the past. During this last month I've learnt quite a bit on how to develop these types of SPAs, yet there was not enough time for me to make it completely functional.*

Riley: *I wrote the code for taking in the URL from the textbox and completing the website scrape by downloading the backend HTML of the sites. I also wrote the code for converting the HTML to a string and completing the MD5 hash and sending that as output to the screen. I created the datatable StoreURL for storing the URLs and used Darren's data table as a template for the StoreHash table. I also connected the SQL data tables to our site and configured the means of storing both the HTML hashes and their URLs in the table. I then wrote the code for the inputted URL hashes to be checked against the hashes in the data table and outputting the decision if the site was updated (not present in the datatable) or not updated (hash was the same and in the data table). I also created the GUI for the manage alerts page and the login/logout page. I also configured the drop down of the site to show all the urls that have been present in the database.*

Darren: *I designed a data-table using T-SQL to store user data. Not all attributes of the table made it into the project, but enough was added to demonstrate the storage of "compressed" website html data to check for updates. I also selected the MD5 hashing algorithm to compress the html data for Riley to help implement. Fixed pathing issues between SQL connection source and generalized parts of Default.aspx so that they would be able to run on everyone's machine.*

Aidan*: Developed the Account and Manager classes to handle the addition of accounts and manage a list of URLs. Worked on datatables and converting the project to the ASP.NET Core framework to function on Macs, in addition to Windows machines, before we ran into too many problems and were forced to revert back to the ASP.NET framework due to time constraints.*

e) ***Video/Presentation***

James: *my video incorporates information about the current framework build and code that I worked on:* ***https://www.youtube.com/watch?v=ErB-62MYc84&feature=youtu.be***

Riley: *my video incorporates the final version of WebSight and a demo: https://youtu.be/v8-Nfyb_hJQ*

7) Link to video

Jame's: ***https://www.youtube.com/watch?v=ErB-62MYc84&feature=youtu.be***
Riley's: *https://youtu.be/v8-Nfyb_hJQ*