Milestone 1 Progress Evaluation

Project Title: Extracting Requirements from Code (ReqEx)

Project Members

- Zachary Bruggen - zbruggen2016@my.fit.edu
- Nicholas Epler - nepler2018@my.fit.edu
- Ivan Hernandez - ihernandez2018@my.fit.edu
- Thomas Morrison - tmorrison2017@my.fit.edu

Faculty Advisor: Dr. Slhoub - kslhoub@fit.edu

Client: Dr. Slhoub, Computer Engineering and Sciences at Florida Institute of Technology

Current Milestone Progress Matrix

Task	**Completion %**	**Zach**	**Nick**	**Ivan**	**Thomas**	**TO DO**
1. Researching aspects of project	100%	25%	25%	25%	25%	None
2. Outline fundamental software methods	80%	30%	10%	30%	10%	Figure out how to pass parsed text to the API and database
3. Create initial GUI mockup	100%	10%	10%	10%	70%	None
4. Create Requirements Doc	100%	25%	25%	25%	25%	Insert nonfunctional requirements when applicable
5. Create Design Doc	100%	30%	10%	30%	30%	None
6. Create Test Plan	100%	30%	20%	30%	20%	Update features and priorities as needed
7. Prepare Presentation	100%	20%	40%	20%	20%	None

Task Discussion

1. Researching

A major portion of this project is going to be text analysis. In particular, a major challenge in this analysis is going to be comments. Since we are trying to generate requirements from code and comments, we will need to be able to determine when comments are relevant and when they are irrelevant. So, we did research on coding standards for different languages, focusing on the standards for comments. Along with this, we did research to understand the different components that the project will use, like databases and APIs.

2. Outline Fundamental Software Methods/Processes

Following the research done into code syntax standards and language standards, we wanted to define a general process that our software should follow. At the highest level, we idealized the process down into 3 steps, uploading a file, scanning the text and generating a requirements report, and then displaying or printing that report. Then, we looked further into that second step, to see what kind of methods we would need to perform that analysis. We then outlined these general methods in pseudocode, and created a simplified flowchart of how the components system should interact. All of this planning was then integrated into our design document, and will be actualized in the code.

3. Create Initial GUI Mockup

Since we are envisioning this to be a standalone software at the moment, we needed GUI to interact with the user. In these mockups, we wanted to get a basic idea of what the user would see when using our product, and make sure that everything was being included according to the requirements. We designed mockups for any action the user could perform, and the effect it would have on the GUI itself. Along with this, since one of our goals is to display text within the GUI, we needed to ensure that the GUI would be able to fit the text such that it is readable. These GUI mockups were then integrated into the design document as initial mockups, and will be updated once they are fully fleshed out.

4. Create Requirements Document

During this milestone, we wanted to outline the formal requirements for our system. During the planning phase, we were doing research on different aspects of the project, while only having a general understanding of the goal we were trying to accomplish. During the first meeting with our client, we started by outlining a general idea of what the system would be able to do. Then, with that we were able to create some very basic mockups of the system. We took these mockups to the next meeting, where we then created a more formal and detailed list of requirements that the system needed to be able to accomplish. From those formal requirements, we were able to design a more concrete architecture, and construct a plan to test the system once it was developed.

5. Create Design Document

After creating the Software Requirements Specification, we needed to formalize our design ideas into the design document. During our work as a team, we would either use whiteboards or our personal computers to draw out design concepts, like the system flow and GUI mockups. All of these were translated into the Design Document, where we were able to convey our ideas for how the system should work. In this document, we placed the mockups of the GUI and the interactions a user could perform. Additionally, we outlined the general path the system would take to generate a list of requirements, along with the system components that would be necessary to interact with each other. The creation of the design document will allow us to determine later on if we are correctly implementing the system's requirements.

6. Create Test Plan

Along with the design document, we also needed to prepare a test plan for our system. In this test plan, we outlined the rationale behind our testing, and the guidelines we would be following. As a quick summary of that document, we would first begin testing a feature for functionality as a standalone object, and then as a feature within the system. Once we could identify that the feature was correctly implemented in the system, we would then test some of the features on their accuracy. Since our system is predicting requirements from text, there is going to be a certain amount of accuracy to that prediction. Therefore, for these features where requirements are being generated, we are going to input sample test files, and see how closely they are able to match our expectations. Overall, the test plan introduces the general idea behind our testing strategy, and the methodology of how we are going to determine the accuracy of the system.

7. Prepare Presentation

Finally, we needed to find a suitable way to present our ideas. In this iteration of the project, we did not prepare any sample programs to demonstrate the full functionality of the system, since we were still in the planning phases of the project. Therefore, we decided to present some of the more important formal requirements of the system. Along with this, we wanted to demonstrate the basic idea of how the system would operate, along with how the individual components would interact with each other in order to generate the list of requirements. Because these ideas are more formalized, in the next iteration of the project we should be able to present some demonstrations of basic interaction with the system.

Teammate Contribution

Zachary Bruggen

Did research into databases, focusing on storing string templates and interacting between multiple databases. Also outlined fundamental software methods, and general process of the system and interactions of the components. Established template for storing items into the database. As for documentation, focused mainly on the creating the Systems Requirements Specification and the Test Plan. Additionally helped with other documentation as needed.

Nicholas Epler

Researched tools to extract requirements from code, made a hello_world program testing parsing code, helped with requirements and other documents, and presentation

Ivan Hernandez

Researched into APIs and which API tools would benefit the project the most. Also looked into previously done studies about text analysis, pattern recognition and matching, and sentence structural models. Helped outline the system architecture and the subprocess necessary for the system. For documentation, mainly worked on the Systems Requirements Specification and Test Plan Documents.

Thomas Morrison

Conducted research to find out a method that could be used to parse the data, and incorporate plain text as well as variables to find the overall requirements of the source code. As well as creating a prototyped GUI to model the final look for the project, and how the user will interact with the code, as well as how the object oriented language will be interpreted by the data. The prototyped GUI is subject to change as the project progresses. As well, I researched a way that the database can interact with the words that are extracted from the code, and in order to make this relation, there is an API key hosted by GitHub Pages that will link the extractor code to the user data.

Next Milestone Plan

Task	**Zach**	**Nick**	**Ivan**	**Thomas**
1. Set up GUI to receive an input file	25%	25%	25%	25%
2. Generate Requirement from a Single Method	25%	25%	25%	25%
3. Output Report to GUI screen, and save as a file	25%	25%	25%	25%

Next Milestone Tasks Discussion

1. Set up GUI to receive an input file

The GUI is used to interact with the system, and allows the system to perform its functions. We need the GUI to be able to accept multiple files at once, or folders that contain multiple files and subfolders and store those in some sort of previously viewed directory. To begin this process though, we want to ensure that input and output will work for a single file

2. Generate requirement from single method

The system is designed to interact with files that contain full source code. To begin to reach the full functionality of our system, we want to start by generating a requirement from a single method. Once we are able to test that this functionality works, we can begin to move into a file with multiple methods, and then a program that uses multiple files. In this process, we will be establishing the database, the connection to the API, and each of the extraction functions that will be used to parse the text

3. Output report to GUI screen, and save as a file

Once we have our generated requirement, we then want to be able to display that output to the screen. With this displayed output, we should also be able to save that output as some sort of file type. Depending on how difficult the file type conversion is, we will either start with just a text file type and implement other types in later iterations, or will implement all proposed file types

Client Meeting Dates

______See faculty advisor meeting dates below

Client Feedback

______See faculty advisor feedback below

Faculty Advisor Meeting Dates

______September 22nd 2021 - Discussed General Requirement Ideas
October 1st 2021 - Formalized Requirement Specifications

Faculty Advisor Feedback

The students were asked to read more about the research problem. This also includes collecting research papers and try to do a literature review on the research problem. Also, several meetings with the advisor were completed to schedule the initial plan of the project. The idea is now to start with a simple design that tackles one issue of the research problem (utilizing method names to extract reqs). Also, the students were able to establish the initial reqs specification document, UI design, and testing plan. As an advisor for this project, I am satisfied with the students' performance so far.

Faculty Advisor Signature: OMITTED

Evaluation by Faculty Advisor

- Faculty Advisor: Detach and return this form to Dr. Chan (HC 214) or email the scores to pkc@csfit.edu
- Score (0-10) for each member: circle a score (or circle two adjacent scores for .25 or write down a real number between 0 and 10)

Zachary Bruggen	0	1	2	3	4	5	5.5	6	6.5	6	7.5	8	8.5	9	9.5	10
Nicholas Epler	0	1	2	3	4	5	5.5	6	6.5	6	7.5	8	8.5	9	9.5	10
Ivan Hernandez	0	1	2	3	4	5	5.5	6	6.5	6	7.5	8	8.5	9	9.5	10
Thomas Epler	0	1	2	3	4	5	5.5	6	6.5	6	7.5	8	8.5	9	9.5	10

Faculty Advisor Signature: ____ Date: __________