Community Board

Software Requirements Specification
Software Project Management & Planning

Team 4:
Brittany
Brandon
George
Edgar

Software Requirements Specifications (SRS)

1. Introduction

1.1 Goals and Objectives

Our goal is to develop an announcement website for people to offer their services such as sales, tutoring, job offering, school-related events. Users will be able to contact the person of the announcement and report inappropriate behavior.

Functional requirements

- The user will register an account and log in. If login is invalid, an error message will display.
- Logged in users can create, update, and delete their announcements.
- Users will be able to view and search announcements made.
- If no announcements are found given the user's criteria, a "Not Found" message will be displayed.
- Users will be able to contact the person of the post via filling an email form. If the user is not logged in, the system will redirect them to the register page.
- Anyone will be able to report inappropriate announcements and will notify administrators.

Non-Functional requirements

- Privacy protection (not sharing private information).
- Quick response time when searching/loading posts.
- Error messages/handling, no crashes of server.
- Users must have an internet connection.

Inverse requirements

- Website doesn't have its own chat system. Customers must fill the contact form and communicate through email.
- Website does not keep track of announcement logs. Once a post has been deleted or been marked as "closed," it will disappear forever.
- Website will not have admin controls to see reports. When an announcement is reported, it will be saved in the database and notify the administrators via email for further action.
- Administrators will review and remove or clear the reported announcement.

1.2 Statement of Scope

Upon opening the website, the user will see the home page in which recent announcements will display. Users will either search for announcements of their interest or register an account in order to be able to post announcements in the board. Once registered, the user will have access to the posting form and be able to publish. On the other hand, any customer interested in any announcement service may click the "contact" button. If not registered, the system will ask the customer to make an account in order to access the contact form. After filling the contact form, an email will be sent to the author of the announcement and the communication goes on from there. Once the customer and author come to an agreement, the author will be able to mark the announcement as “Closed,” which will be deleted after some time.

1.3 Software Context

Due to the COVID-19 pandemic, bulletin boards are not in use anymore. Schools closed, people are at home, and school events/clubs are not as known. Very few events are sent through the school emails which not many people check. For those reasons, our team has decided to make these drastic changes alive again.

The purpose of this website is to provide our community a virtual bulletin board (similar to what OCC had in the walls before the pandemic). This is highly useful for those looking to provide their service such as tutoring, those looking for a job, club, or event to join/apply, or even people who might be graduating and want to sell their books or school supplies for a cheaper price.

1.4 Major Constraints

- Internet access required for the whole system.
- Database access.
- REST API calls using HTTP protocol.

2.0 Usage Scenario

2.1 User Profiles

User creates an account with name, username, email, and password. Users post as many announcements of services, or events on the message board. Customers can browse the board for items of their interest, then create an account with their information to email a message to the user. Anyone can report inappropriate announcements.

2.2 Use Cases

Home Page

Actor	Role	Description
User/Customer	Poster/Visitor	Views announcements already posted
Program (API)	Data Communication	Handles HTTP request/response
Database	Data Storage	Extracts announcements to view

User Case Interaction

Predecessor: None. This is the landing page. Everyone has access to this page regardless of having an account or not.

Successor: Register Page.

Register Page

Actor	Role	Description
User/Customer	Poster/Visitor	Enters credential information
Program (API)	Data Communication	Handles HTTP request/response
Database	Data Storage	Registers user credentials

User Case Interaction

Predecessor: Home page.

Successor: Login Page.

Log in

Actor	Role	Description
User/Customer	Poster/Visitor	Enters credential information
Program (API)	Data Communication	Handles HTTP request/response
Database	Data Storage	Extracts data for authentication

User Case Interaction

Predecessor: Register Page. The user must register an account first.

Successor: Any other use case.

Create Announcement

Actor	Role	Description
User	Poster	Creates announcement using form
Program (API)	Data Communication	Handles HTTP request/response

Database	Data Storage	Saves and stores announcement

User Case Interaction

Predecessor: Login. The user must be logged in.
Successor: Any other use case.

Update Announcement

Actor	Role	Description
User	Poster	Updates the announcement.
Program (API)	Data Communication	Handles HTTP request/response
Database	Data Storage	Updates the announcement

User Case Interaction

Predecessor: Login & Create Announcement. The user must be logged in and have created an announcement.
Successor: Any other use case.

Delete Announcement

Actor	Role	Description
User	Poster	Deletes the announcement
Program (API)	Data Communication	Handles HTTP request/response
Database	Data Storage	Deletes the announcement

User Case Interaction

Predecessor: Login & Create Announcement. The user must be logged in and have created an announcement.
Successor: Any other use case.

Contact/Email Announcement Author

Actor	Role	Description
Customer	Visitor	Fills contact form and sends a

		message to the author.
Email API	Email Communication	Notifies post author about customer message regarding the announcement.

User Case Interaction

Predecessor: Login. The user must be logged in in order to contact the author.
Successor: Any other use case.

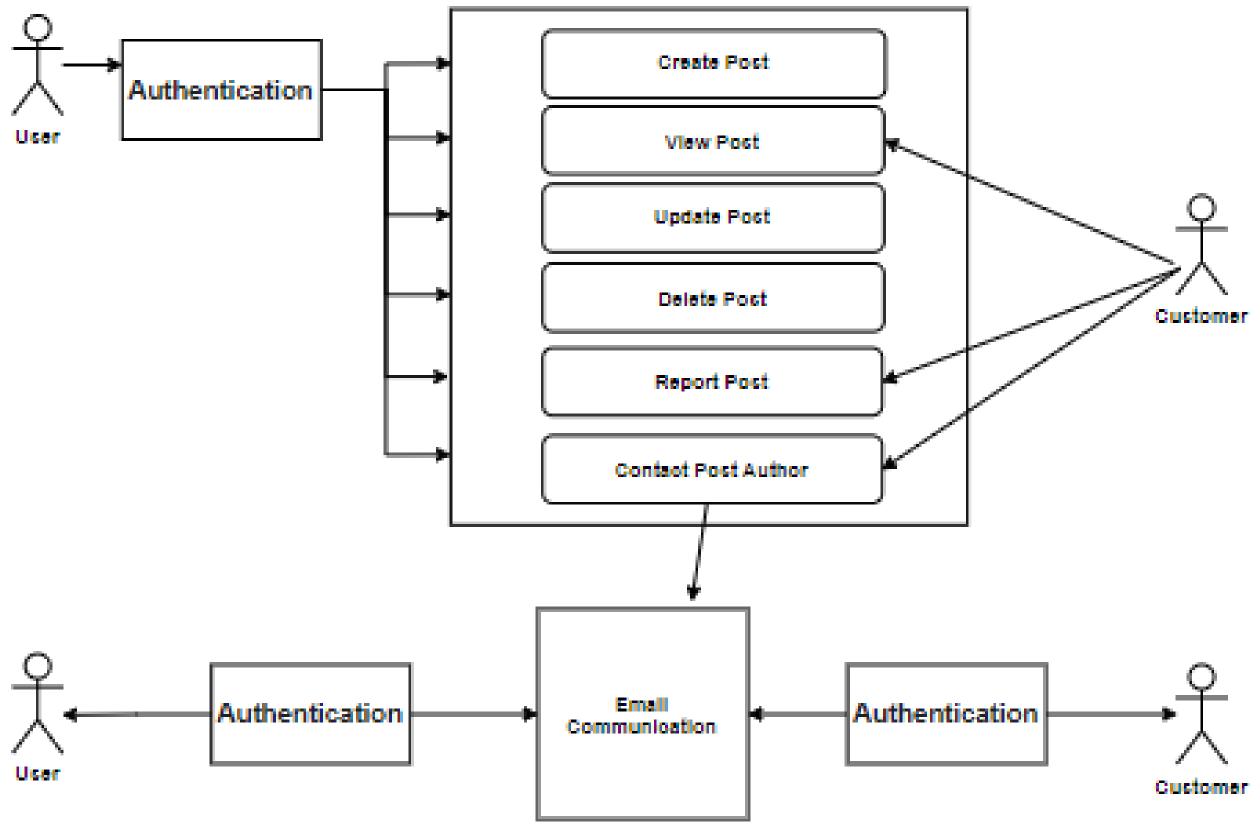
Report Announcement

Actor	Role	Description
User/Customer	Poster/Visitor	Reports announcement
Program (API)	Data Communication	Handles HTTP request/response
Database	Data Storage	Saves report information
Email API	Email Communication	Notifies administrators about the report.

User Case Interaction

Predecessor: Home Page. The user must be on the home page and may not be registered. Anyone can report.
Successor: Any other use case.

2.3 Use Case Diagram Considerations



4.2 Software Interface Description

4.2.1 External Machine Interfaces

This website does not plan to communicate with any external machine.

4.2.2 External System Interfaces

- The frontend (what users can see) will be connected to the backend system through REST API (Application Programming Interface) requests and responses using the HTTP Protocol.
- Email API. Will be used to send email.

4.2.3 Human Interface

The human interface will allow:

- Users to create and modify announcements as desired.
- Customers to contact the author of the post completing the email form.
- Any person to report a misleading post.

5.0 Behavioral Model and Description

5.1.1 Events

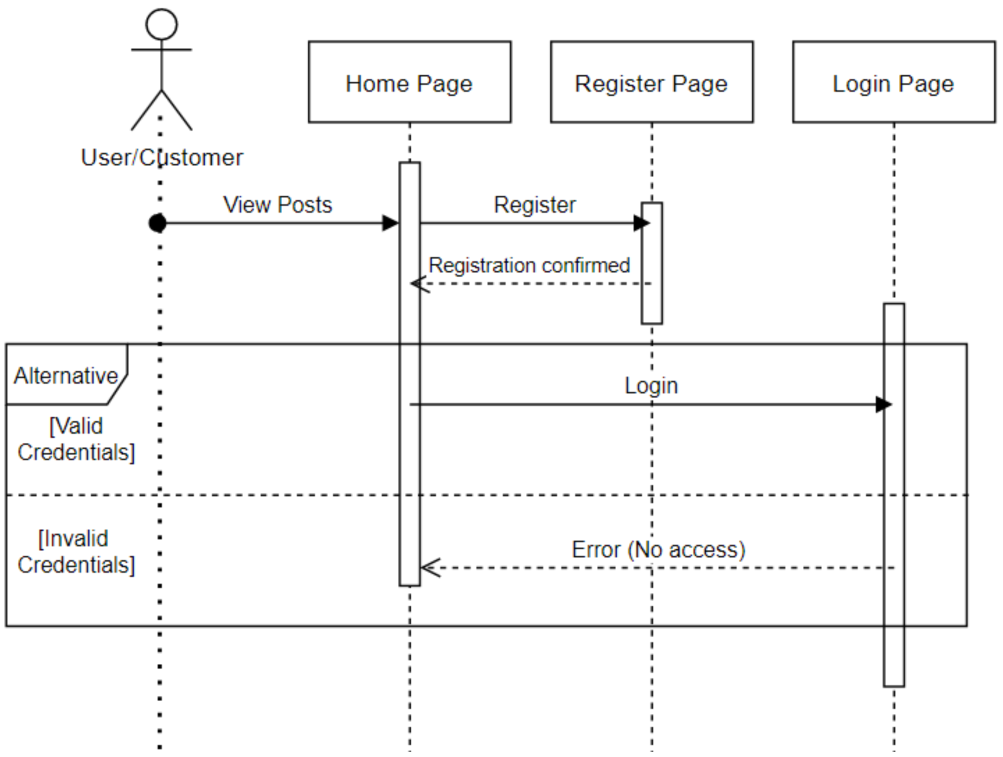
- Landing Page (Home Page).
  - Announcements are loaded
- Register Page.
- Login Page.
- Create Announcement Page.
- Update/Delete Announcement.
- Contact Author.
- Report Announcement.

5.1.2 States

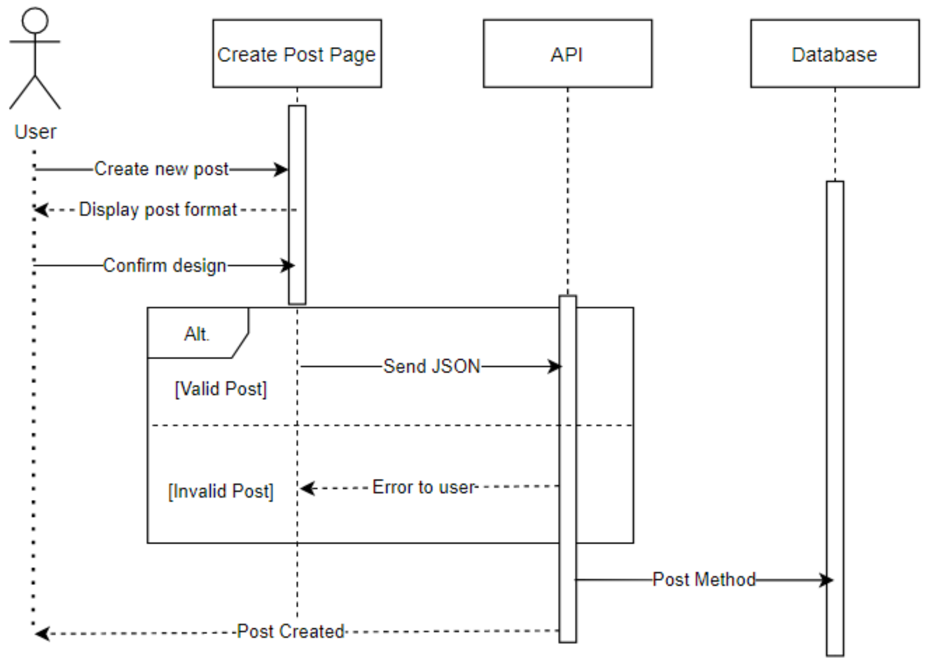
- User Authentication.
- Reported Announcement (Flagged).
- Updated/Deleted Announcement.

5.2 Sequence Diagrams

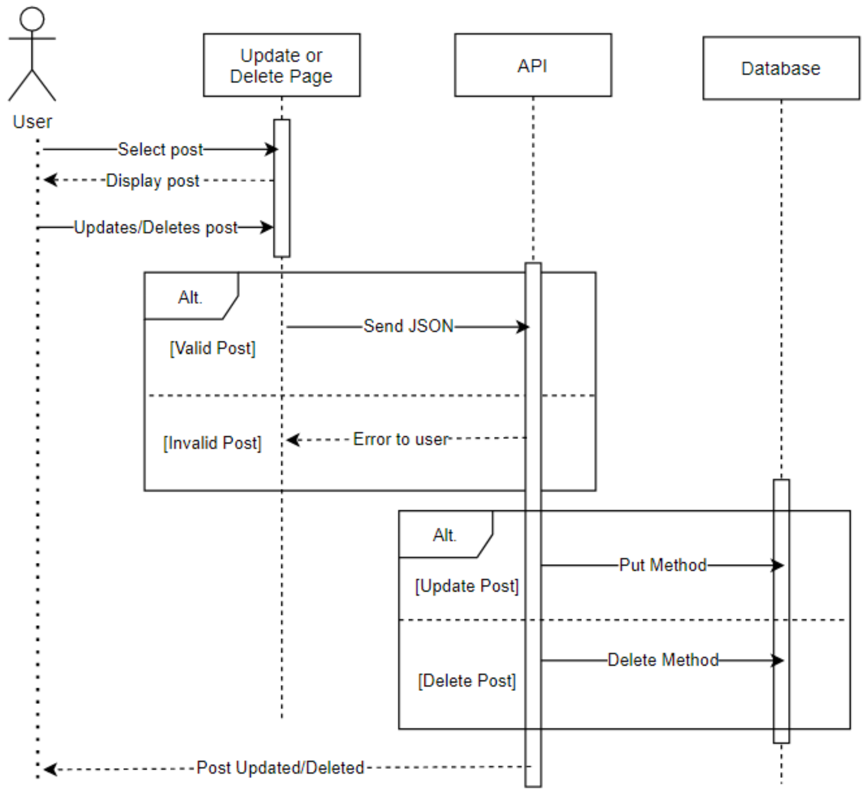
Home Page & Registration/Login



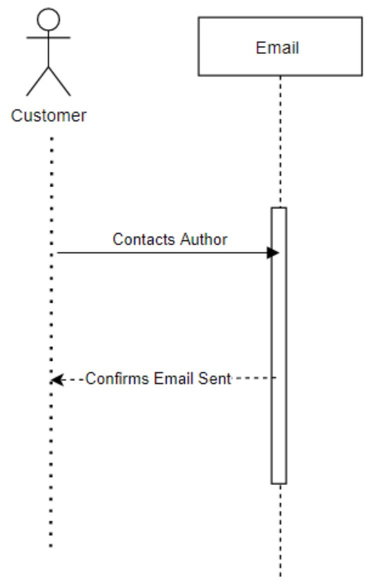
Create Announcement



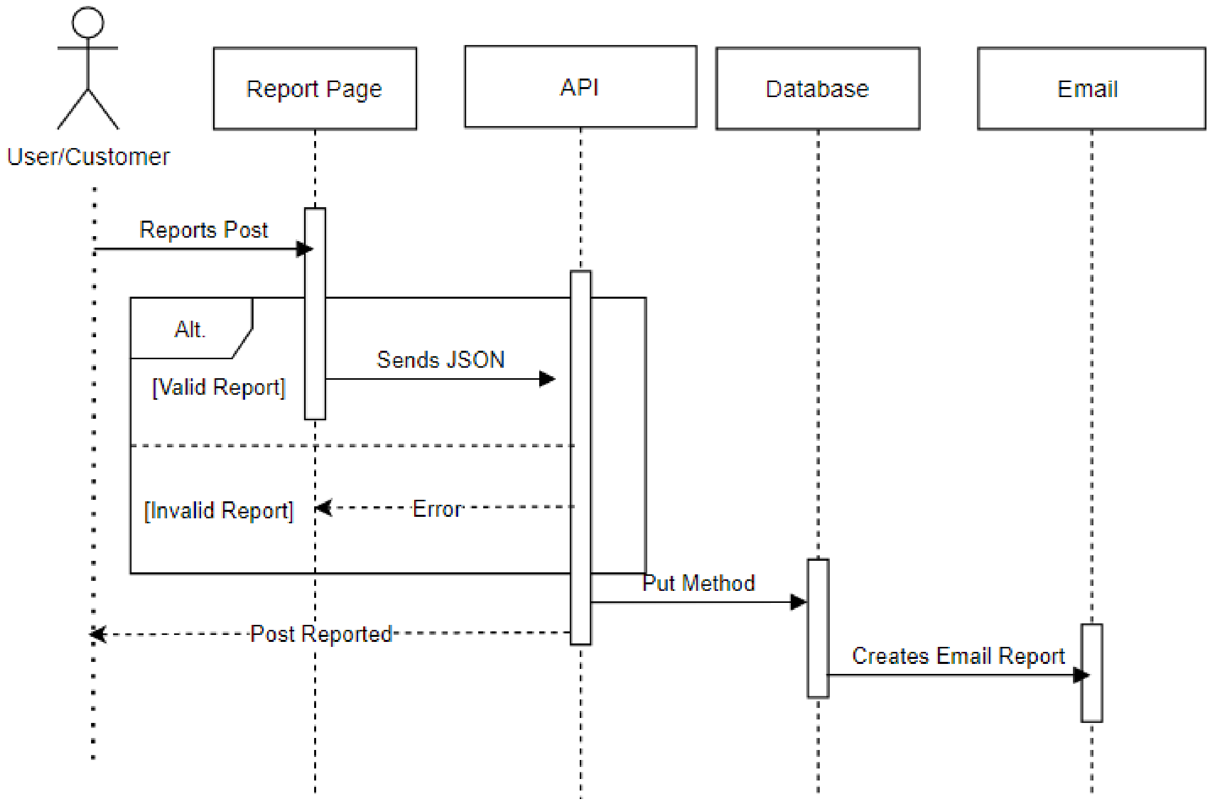
Update/Delete Announcement



Email Announcement Author



Report Announcement



6.0 Restrictions, Limitations, and Constraints

- Website load time may depend on the user's internet speed
- Web browsers limited to: Edge, Safari, Chrome, Firefox, Opera.
- Database Access Performance: Entity Framework can be slower when accessing DB.
- Server hosting plans may need to be high to support better performance when accessing the database with Entity Framework in production. This project is made by students with not much experience. Therefore, the server will be optimized to the best of our ability.

7.0 Validation Criteria

7.1 Classes of Tests

- Unit Testing: test that each function performs according to specifications.
- Functional Testing: test website components work as expected.
- Performance Testing: test api calls response times and database access.
- User Interface Testing: test UI usability.

7.2 Expected software response

The software will pass or fail the tests.

7.3 Performance bounds (i.e. app crashes expected, etc.)

- Server performance issues are expected.
- Server crashes are definitely expected during development.
- RAM/in-memory database access performance/speed issues expected.

8.0 Appendices

Configuration Management

To collaborate in our team, we are using:

- Google docs:
  - Reports for SRS/SPMP/Final Report.
  - Team notes for later use.
- Github/Git:
  - Frontend design & implementation code.
  - Backend design & implementation code.
- Zoom & Discord:
  - Communication.
  - Project planning.
- Draw.io:
  - Designing:
    - UML diagrams.
    - ERD diagram.
    - Use-case & Sequence diagrams.

User Project Resources

Minimal Hardware Resources

Development:

- Processor: x86 or x64.
- RAM: 512 MB - 1 GB (minimum), 4GB or higher (recommended)
- Hard disk: 2GB (minimum), 4GB or higher (recommended).

User:

- Any 32-bit or 64-bit Operating System.

Minimal Software Resources

Development:

- Visual Studio
- VS Code (Optional)
- .NET Core 3.1
- ASP.NET Core 3.1
- SQL Server Management Studio
- Postman

User:

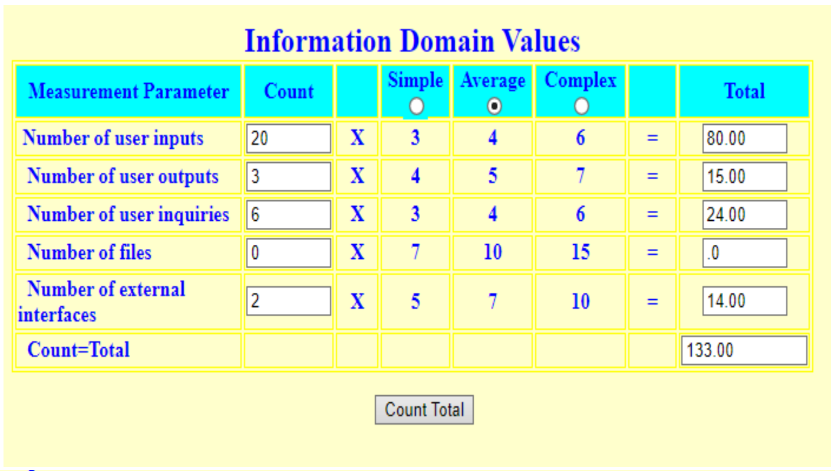
- Any web browser (Internet Explorer NOT recommended)

References

http://www.rspa.com/docs/Reqmspec.html

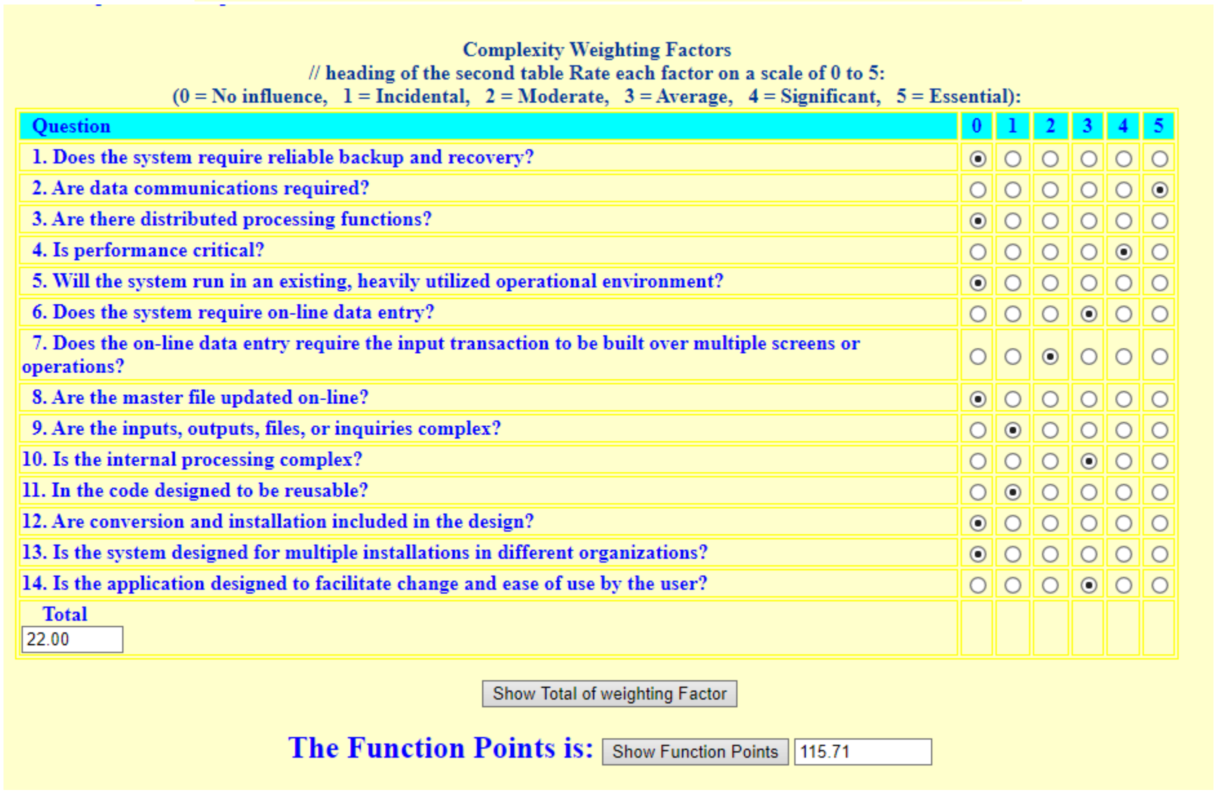
Software Project Management & Planning (SPMP)

2.0 Project Estimates



Information Domain Values

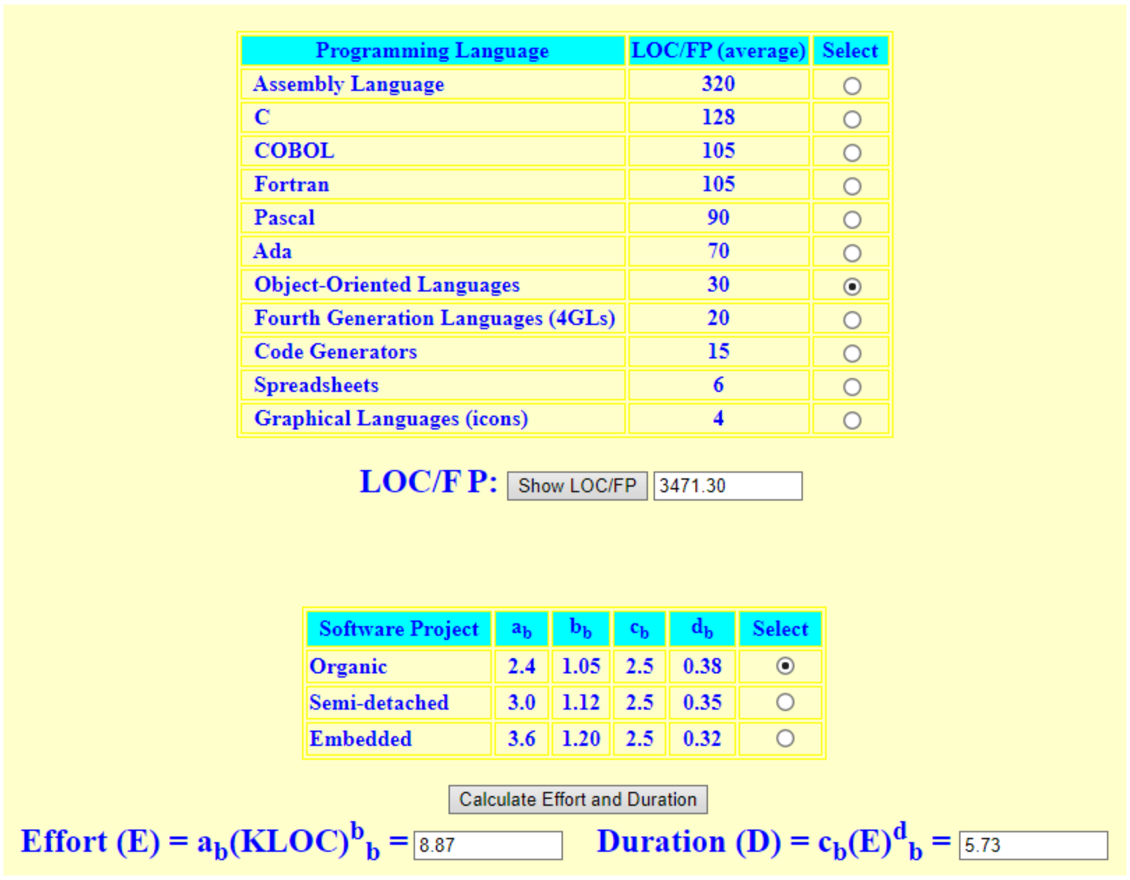
Measurement Parameter	Count		Simple ○	Average ●	Complex ○		Total
Number of user inputs	20	X	3	4	6	=	80.00
Number of user outputs	3	X	4	5	7	=	15.00
Number of user inquiries	6	X	3	4	6	=	24.00
Number of files	0	X	7	10	15	=	.0
Number of external interfaces	2	X	5	7	10	=	14.00
Count=Total							133.00



Complexity Weighting Factors
// heading of the second table Rate each factor on a scale of 0 to 5:
(0 = No influence, 1 = Incidental, 2 = Moderate, 3 = Average, 4 = Significant, 5 = Essential):

Question	0	1	2	3	4	5
1. Does the system require reliable backup and recovery?	●	○	○	○	○	○
2. Are data communications required?	○	○	○	○	○	●
3. Are there distributed processing functions?	●	○	○	○	○	○
4. Is performance critical?	○	○	○	○	●	○
5. Will the system run in an existing, heavily utilized operational environment?	●	○	○	○	○	○
6. Does the system require on-line data entry?	○	○	○	●	○	○
7. Does the on-line data entry require the input transaction to be built over multiple screens or operations?	○	○	●	○	○	○
8. Are the master file updated on-line?	●	○	○	○	○	○
9. Are the inputs, outputs, files, or inquiries complex?	○	●	○	○	○	○
10. Is the internal processing complex?	○	○	○	●	○	○
11. In the code designed to be reusable?	○	●	○	○	○	○
12. Are conversion and installation included in the design?	●	○	○	○	○	○
13. Is the system designed for multiple installations in different organizations?	●	○	○	○	○	○
14. Is the application designed to facilitate change and ease of use by the user?	○	○	○	●	○	○
Total 22.00						

The Function Points is: Show Function Points 115.71



Programming Language	LOC/FP (average)	Select
Assembly Language	320	○
C	128	○
COBOL	105	○
Fortran	105	○
Pascal	90	○
Ada	70	○
Object-Oriented Languages	30	◉
Fourth Generation Languages (4GLs)	20	○
Code Generators	15	○
Spreadsheets	6	○
Graphical Languages (icons)	4	○

LOC/F P: Show LOC/FP 3471.30

Software Project	a_b	b_b	c_b	d_b	Select
Organic	2.4	1.05	2.5	0.38	◉
Semi-detached	3.0	1.12	2.5	0.35	○
Embedded	3.6	1.20	2.5	0.32	○

Calculate Effort and Duration

Effort (E) = $a_b(\text{KLOC})^{b}{}_{b}$ = 8.87   Duration (D) = $c_b(E)^{d}{}_{b}$ = 5.73

Project Resources

People

- Brittany
- Brandon
- George
- Edgar

Hardware

- Laptops, Desktops

Software

- C# with ASP.NET Razor
- SQL Server Management Studio
- Postman
- Draw.io
- Visual Studio
- Visual Studio Code
- Google Docs
- Github/Git
- Discord

3.0 Risk Management

Risk Table

Risk Name	Impact	Mitigation	Contingency Plan
Absenteeism (Sick, COVID)	Medium	If there is time, let the team member complete when available.	Other team members may assist.
Difficulty Using unfamiliar software/tools/ technologies	Medium	Read documentation, take a small crash course, trial and error, practice.	Use familiar software.
Project Changes	Medium	Only necessary changes are the ones to be allowed.	Project documents and design would have to be changed and redesigned.
Time Concerns	Low	Team will have to work on tasks faster, which implies more hours dedicated.	Consider a revaluation of tasks' length or size.
Errors/Exceptions /Crashes	Medium	Error handling while coding.	Debug or redesign code structure.

4.0 Project Schedule

Project Tasks and Timeline Chart

Task ID	Task	Completion Date	Team Member(s)	Predecessor	Successor
A	SRS/SPMP	8 Feb 2021	Everyone		ALL
B	Prototype Presentation	15 Feb 2021	Everyone	A	C
C	SDD	22 Feb 2021	Brittany, Edgar	A	D
D	Backend & Frontend	5 April 2021	Edgar	A	E
E	Testing	15 April 2021	Everyone	D	F
F	Final Report Guide	19 April 2021	Brandon, George	E	G
G	Final Presentation	19 April 2021	Everyone	ALL	

5.0 Staff Organization

Team Structure

Brittany	Database Design (ERD), Diagrams
Brandon	Documentation, Reports, Beta Tester
George	Diagrams, Documentation, Reports
Edgar	FullStack Programming (C#, API design, database implementation, HTML, CSS, JavaScript)

Management Reporting and Organization

Tasks are divided among team members. However, we will be contributing to other tasks as necessary (as team members gain experience in coding, they may contribute to the source code as well). We will be keeping in contact and meeting with each other to discuss progress in zoom and discord.

6.0 Appendix

http://www.rspa.com/docs/Projectplan.html
http://groups.umd.umich.edu/cis/course.des/cis525/js/f00/gamel/cocomo.html