**3.1**

**Start Date:** January 10th,2022

**End Date:** November 17th, 2023

When deciding how long the project would take, the calculations were based on the employees having the weekends off. The reason for the start date being January 10th is because that is when the holiday season is over and to lower the cost the project will be taking place during the time of the year when there are no paid holidays. When it came to deciding how long it would take for the project to be done, according to ***Proxity*** the average project takes 12 months. With our projects with all the API’s that we are planning to use we can assume that the time for actual development will be shorter. If gathering all the requirements and designing the UX/UI takes a month to and month to a half, implementation for three bare bone versions takes two and a half months assuming each version is going to take two to three weeks. After the implementation, the verification of the application will take the longest where if we were to develop a minimum maintenance application. According to Proxitythis stage will take 5 months which brings the total to 10 months. And the following days left will be for maintenance after the launch of the application. The reason the project only gives an extra week is so that there is no PTO (paid time off) for the developers that were hired specifically for this project. All the assumptions are made with extra slack time in mind that might not be needed.

‌

**3.2**

**For this project we have decided to use the FP method to estimate the Effort.**

1 – Determine function category

Number of user inputs – **8**

Number of user outputs – **4**

Number of user queries – **3**

Number of data tables and relational files – **10**

Number of external interfaces – **4**

2 – Determine complexity

Number of user inputs – **Average**

Number of user outputs – **Simple**

Number of user queries – **Complex**

Number of data tables and relational files – **Simple**

Number of external interfaces – **Average**

3 – Compute Gross Function Point (GFP)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Function Category | Count | Complexity | | | Count & complexity |
| Simple | Average | Complex |
| 1 | **Number of user inputs** | **8** | 3 | 4 | 6 | **32** |
| 2 | **Number of user outputs** | **4** | 4 | 5 | 7 | **16** |
| 3 | **Number of user queries** | **3** | 3 | 4 | 6 | **18** |
| 4 | **Number of data tables and relational files** | **10** | 7 | 10 | 15 | **70** |
| 5 | **Number of external interfaces** | **4** | 5 | 7 | 10 | **28** |
| **GFB** | | | | | | **164** |

4 – Determine the PC

(1) Does the system require reliable backup and recovery? **2**

(2) Are data communications required? **3**

(3) Are there distributed processing functions? **2**

(4) Is performance critical? **2**

(5) Will the system run in an existing, heavily utilized operational environment? **3**

(6) Does the system require online data entry? **4**

(7) Does the online data entry require the input transaction to be built over multiple screens or operations? **4**

(8) Are the master files updated online? **3**

(9) Are the inputs, outputs, files, or inquiries complex? **5**

(10) Is the internal processing complex? **3**

(11) Is the code designed to be reusable? **2**

(12) Are conversion and installation included in the design? **1**

(13) Is the system designed for multiple installations in different organizations? **5**

(14) Is the application designed to facilitate change and ease of use by the user? **1**

5 – Compute processing complexity adjustment (PCA)

**PCA = 0.65 + 0.01(2 + 3 + 2 + 2 + 3 + 4 + 4 + 3 + 5 + 3 + 2 + 1 + 5 + 1)**

**PCA = 1.05.**

6 – Compute function point (FP) using the formula

FP = GFB \* PCA

FP = 164 \* 1.05

FP = 172.2

**Conclusion on Effort -**

E = FP / productivity

E = 172.2 / 60 = 2.87 ≈ **3 person-weeks.**

D = E / team size

**Conclusion on Cost-**

D = 3 / 2 = 1.5 ≈ **2 weeks**.

If average productivity is 155 pw.

Pw = per week.

Estimated lines of code = 2 \* 620

Estimated lines of code = 1240;

Cost = LOC \* avg Cost Per Line of Code

Cost = 1240 \* 15.415 = **$19,114.6**

**Conclusion on Price-**

The price to the customers will be free and the profit will be made off the ads, however for the premium version the price will be $9.99

(For 3.1)

[**https://proxify.io/articles/how-long-does-it-take-to-develop-an-app#don-t-rely-on-rough-app-development-time-estimates**](https://proxify.io/articles/how-long-does-it-take-to-develop-an-app#don-t-rely-on-rough-app-development-time-estimates)

[1]

“How to correctly estimate your app development timeline,” *proxify.io*. <https://proxify.io/articles/how-long-does-it-take-to-develop-an-app#don-t-rely-on-rough-app-development-time-estimates> (accessed Nov. 10, 2022).

(For 3.2)

<https://vladnevzorov.wordpress.com/2011/01/31/how-much-does-it-cost-to-develop-a-line-of-code-sloc/#:~:text=With%20this%20approach%20given%20an,be%20between%20%2412.33%20and%20%2418.5>.

[1]

“How much does it cost to develop a line of code?,” *Vlad Nevzorov’s blog*, Jan. 31, 2011. <https://vladnevzorov.wordpress.com/2011/01/31/how-much-does-it-cost-to-develop-a-line-of-code-sloc/#:~:text=With%20this%20approach%20given%20an> (accessed Nov. 10, 2022).

‌