**Batch: A1**

**Roll No.: 1911004, 1911005, 1911012**

**Experiment / assignment / tutorial No.2\_**

|  |
| --- |
| **TITLE: Project Metric estimations for Mini Project** |

**AIM:** To enable the students learn different techniques for performing software size and cost estimation

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Expected Course outcome of Experiment:**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Books/ Journals/ Websites referred:**

1. Roger Pressman, “Software Engineering”, sixth edition, Tata McGraw Hill.
2. <http://sunset.usc.edu/csse/research/COCOMOII/cocomo_main.html>
3. <http://sunset.usc.edu/research/COCOMOII/expert_cocomo/expert_cocomo2000.html>
4. <http://groups.umd.umich.edu/cis/course.des/cis525/js/f00/gamel/cocomo.html>
5. <http://softwarecost.org/tools/COCOMO/>

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Pre Lab/ Prior Concepts:**

Software projects have tendency of going past their deadline, going over budget, or both. The problem lies in the estimation of the amount of effort required for the development of a project. The cost estimation is usually dependent upon the size estimate of the project, which may use lines of code or function points as metrics. There are several different techniques for performing software cost estimation, including expert judgement and algorithmic models. Estimation by expert judgement is a common way of estimating the effort required for a project. Unfortunately, this method of estimation does not emphasize re-estimation during the project life cycle, which is an important part of project tracking, because it allows the estimates to be improved during the project life cycle. The quality of a cost estimation model is not so much attributed to the initial estimate, but rather the speed at which the estimates converges to the actual cost of the project. COCOMO is a popular algorithmic model for cost estimation whose cost factors can be tailored to the individual development environment, which is important for the accuracy of the cost estimates. More than one method of cost estimation should be done so that there is some comparison available for the estimates. This is especially important for unique projects. Cost estimation must be done more diligently throughout the project life cycle so that in the future there are fewer surprises and unforeseen delays in the release of a product.

**Estimation of size and cost of the developing project is required for the following major decision situations**

* Financial decisions involving a software development effort
* Setting project budgets and schedules as a basis for planning and control
* Deciding on or negotiating tradeoffs among software cost, schedule, functionality, performance or quality factors
* Making software cost and schedule risk management decisions
* Deciding which parts of a software system to develop, reuse, lease, or purchase
* Making legacy software inventory decisions: what parts to modify, phase out, outsource, etc
* Deciding how to implement a process improvement strategy, such as that provided in the SEI CMM

Defining Cost estimation:

Cost estimation can be defined as the approximate judgement of the costs for a project. Cost estimation will never be an exact science because there are too many variables involved in the calculation for a cost estimate, such as human, technical, environmental, and political. Furthermore, any process that involves a significant human factor can never be exact because humans are far too complex to be entirely predictable. Furthermore, software development for any fair-sized project will inevitably include a number of tasks that have complexities that are difficult to judge because of the complexity of software systems.

Cost estimation is usually measured in terms of effort. The most common metric used is person months or years (or man months or years). The effort is the amount of time for one person to work for a certain period of time. It is important that the specific characteristics of the development environment are taking into account when comparing the effort of two or more projects because no two development environments are the same. A clear example of differences in development environments are the amount of time people work in different countries; the typical workweek in North America is 40 hours per week, while in Europe the typical workweek is 35 hours per week. Thus, when comparing a project from North America with a project from Europe, a conversion factor would have to be used to all for an accurate comparison. Different variables can be used for cost estimation, which leads to a difficulty when comparing projects if standard models or tools are not used. For example, a cost estimate can include factors from management, development (e.g., training, quality assurance), and other areas specific to an organization.

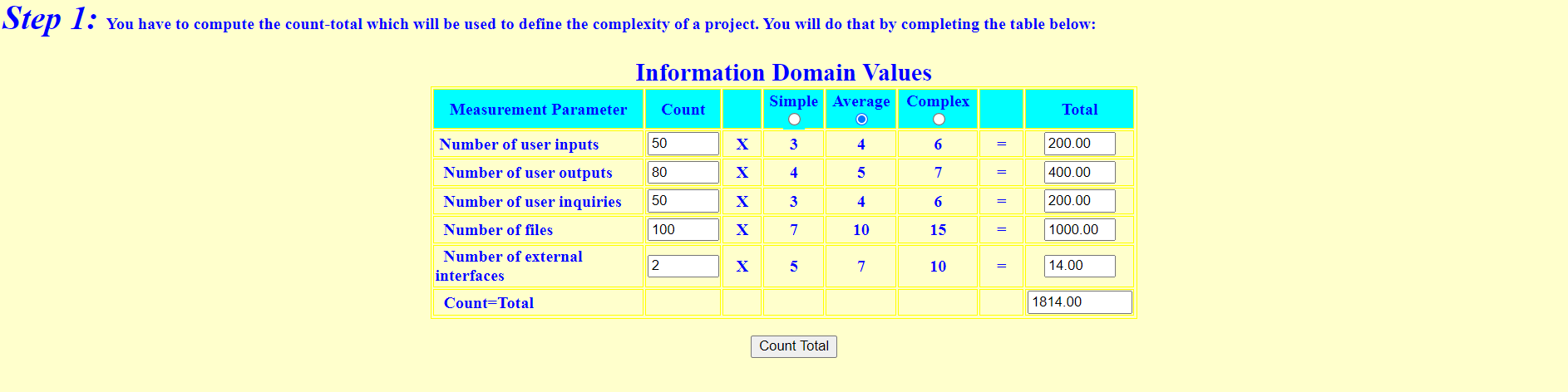
Estimator:

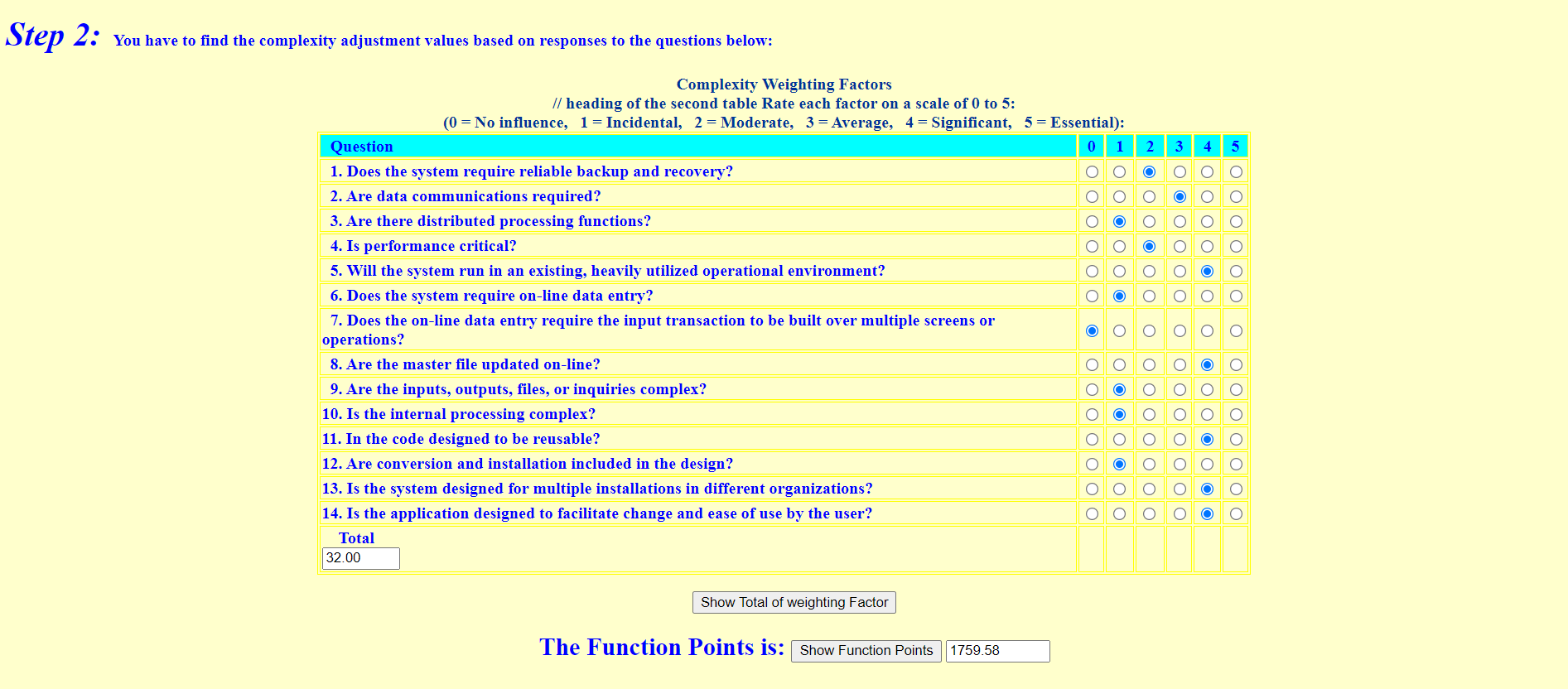
The people who do the cost estimates could be either directly or indirectly responsible for the implementation for a project, such as a developer or manager, respectively. Someone who has knowledge of the organization and previous projects could use an analogy-based approach to compare the current project with previous projects, which is a common method of estimation for small organizations and small projects. The historical data is often limited to the memory of the estimator. In this case, the estimator would need to be experienced and would likely have been with the company for awhile.

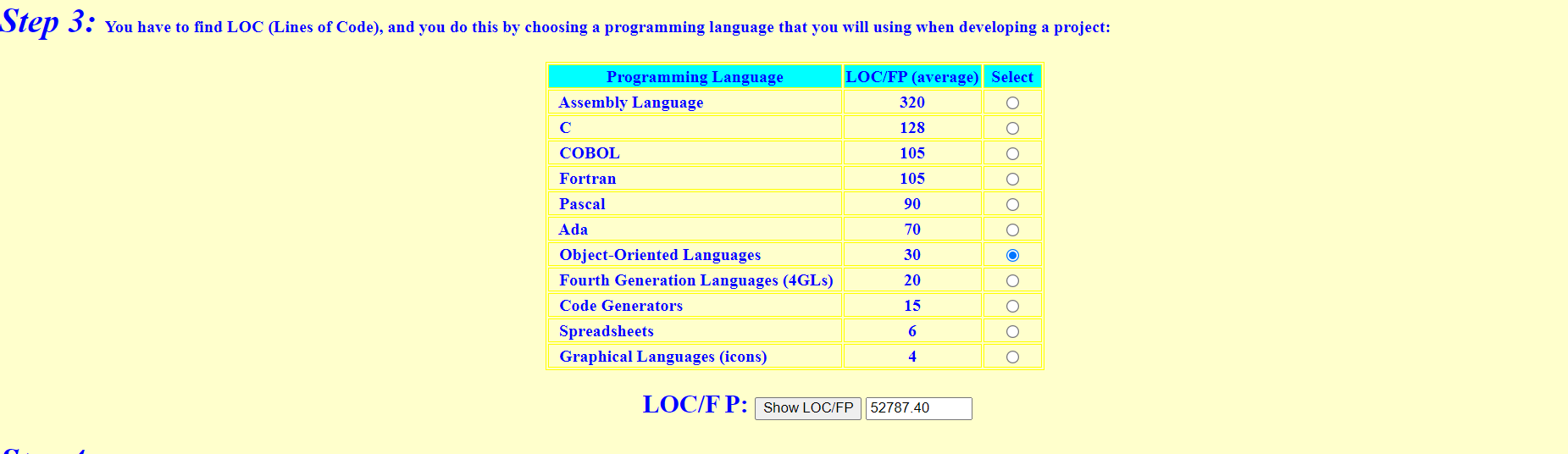
Some people believe it is better if the estimates are done by outsiders so that there is less chance of bias. It is true that people outside an organization will likely have to deal with fewer company politics than people within the organization. For example, the developer for a company may want to please the manager and so give an estimate that is overly-optimistic. The disadvantage of having an outside estimate is that the person would have less knowledge of the development environment, especially if the person is from outside the company. An empirical method of estimation would then be required, such as the Constructive Cost Model (COCOMO. Empirical methods of estimation can be used by all types of estimators. There may be some resistance to using an empirical method of estimation because there may be some question on whether a model could outperform an expert. People who are accurate estimators are rare in our experience, and so it is best to get the opinion of several people or tools.

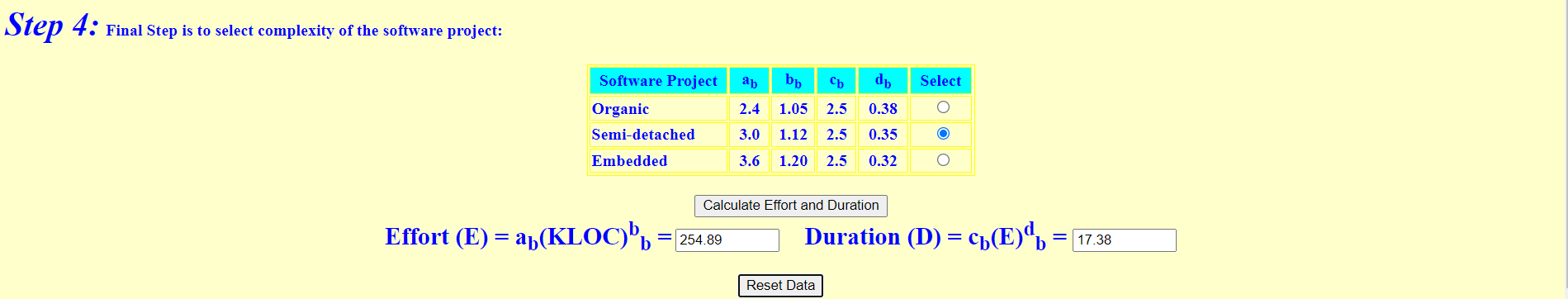
**Cost estimation using different COCOMO models:**

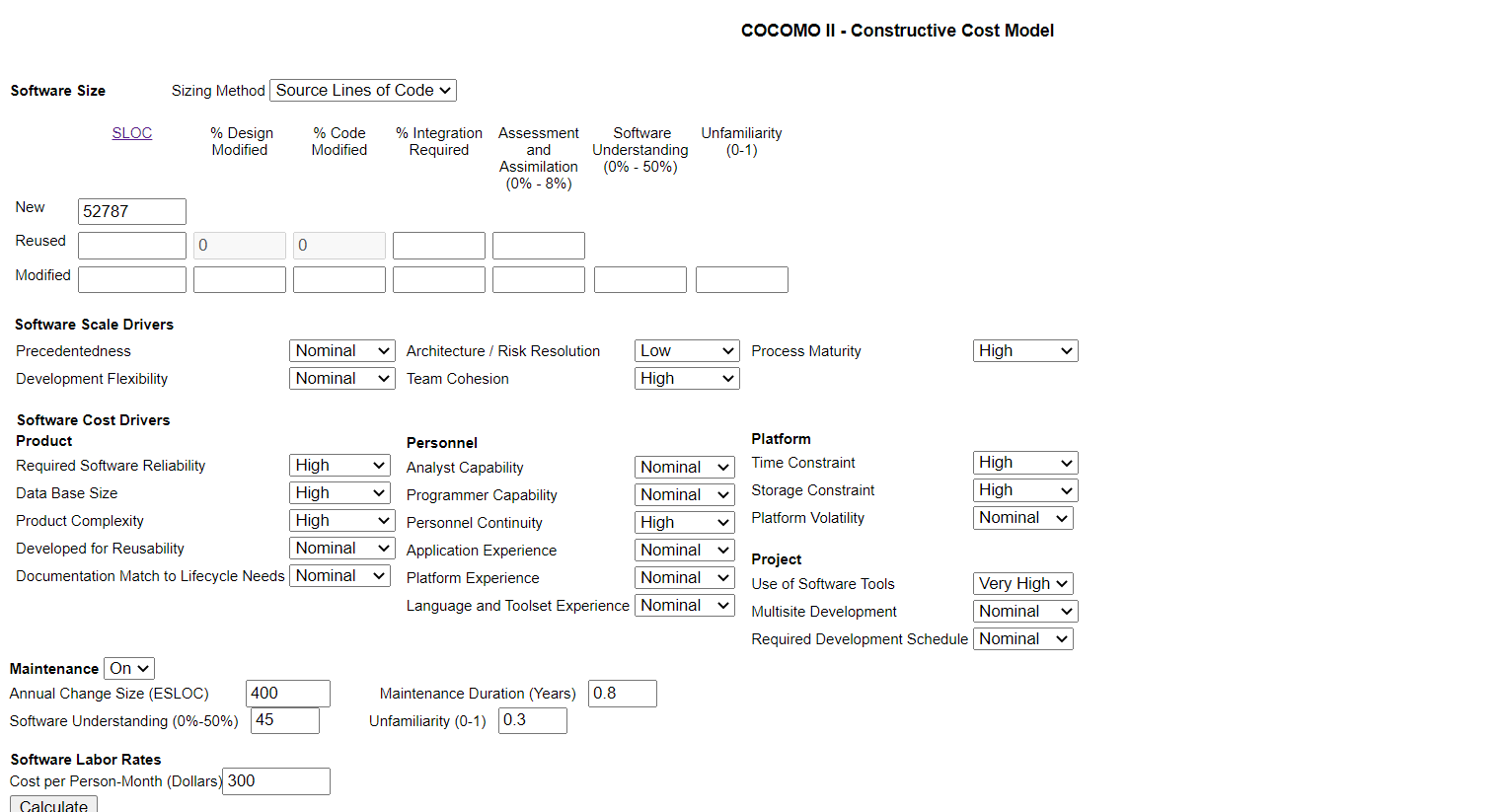
|  |  |  |
| --- | --- | --- |
| **Software Scale Drivers** | **Value** | **Justification** |
| Precedentedness | Nominal | We have not implemented any project of this kind prior to this one, neither we have worked on a project with such complexity and this particular tech stack. |
| Architecture / Risk Resolution | low | The Architecture/Risk Resolution is low since the students and faculties from the institute itself will be accessing the project. |
| Process Maturity | High | Due to nominal precedenteness in the project the extent of definiteness of the processes is High. |
| Development Flexibility | Nominal | The development Flexibility will be nominal since the app will be dependent on the internet connection of the student and if there is low connectivity in particular areas the student may not be able to access the content. |
| Team Cohesion | High | The team cohesion is bound to be high since all the members developing the product are under the same branch hence the time devoted towards the development of the product will be more. |
|  |  |  |
| **Software Cost Drivers** |  |  |
| **Product** |  |  |
| Required Software Reliability | High | The required software reliability is high as the project will be host on reliable server of the college domain which have very low downtime. |
| Data Base Size | High | Since we are aiming at developing this system for KJSCE, and considering the vast syllabus of different branches the database size needed will be high. |
| Product Complexity | High | The product Complexity will be high since there are multiple features and a huge chunk of people will be access the product at the same time. |
| Developed for Reusability | Nominal | The idea of project allows it to be reused for any organisation a e-learning system with some minor shifts. |
| Documentation Match to Lifecycle Needs | Nominal | Will be nominal since we cannot cover all the aspects and have solution to problems pertaining to network issues etc. |
| **Personnel** |  |  |
| Analyst Capability | Nominal | Our project is not focused or based on data as a main stream. Hence the value is nominal. |
| Programmer Capability | Nominal | Since we are still learning various technologies which are required to build this project, the programmer capability is nominal. |
| Personnel Continuity | High | We would be working on the project continuously during the span of about 6 months. |
| Application Experience | Nominal | Is nominal as all our team members do not have much exposure of all the technologies which are required. |
| Platform Experience | Nominal | Is nominal since we are very much familiar with the platform used to develop the application. |
| Language and Toolset Experience | Nominal | Is also nominal considering we are still in the learning stage. |
| **Platform** |  |  |
| Time Constraint | High | The time constraint will be high since we are not yet familiar with some of the technologies and it will take some time to get familiar with it. |
| Storage Constraint | High | The storage Constraint will be high considering the huge amount of data which will be accessed by so many people. |
| Platform Volatility | Nominal | The platform is not much volatile at this particular position in time but seeing the current advancements in the field of technology, we speculate that there might be a better, more efficient way of carrying out the task. |
| **Project** |  |  |
| Use of Software Tools | Very High | As we are developing a system which is totally online and considering every person will have a mobile handset. So this decreases our hardware cost to just a server depending on the requirements. |
| Multisite Development | Nominal | This value is Nominal because the product does not demand a team on the client side. The developer team can handle both the development as well as the client communication. |
| Required Development Schedule | Nominal | Will be nominal since only a brief schedule will be shared with the customer or company to which the product is going to be delivered. |
|  |  |  |
| **Maintenance** |  |  |
| Annual Change Size (ESLOC) | 400 | This is a pure guess as it would not be any estimated answer for this. This value is mainly determined by the situation at the time of maintenance. |
| Maintenance Duration (Years) | 0.8 | This would be a standard time for the maintenance. |
| Software Understanding (0%-50%) | 45 | Software must be understood fairly for the maintenance. |
| Unfamiliarity (0-1) | 0.3 | If we have understood the software completely there is always a possibility of introduction of a new thing whenever we enhances in the direction of development. |
|  |  |  |
| **Software Labour Rates** |  |  |
| Cost per Person-Month (Dollars) | 300 | This value is taken considering that what we have thought of the project to be at the end. If we consider that the project is delivered on time with all necessary features then this value can be justified. |

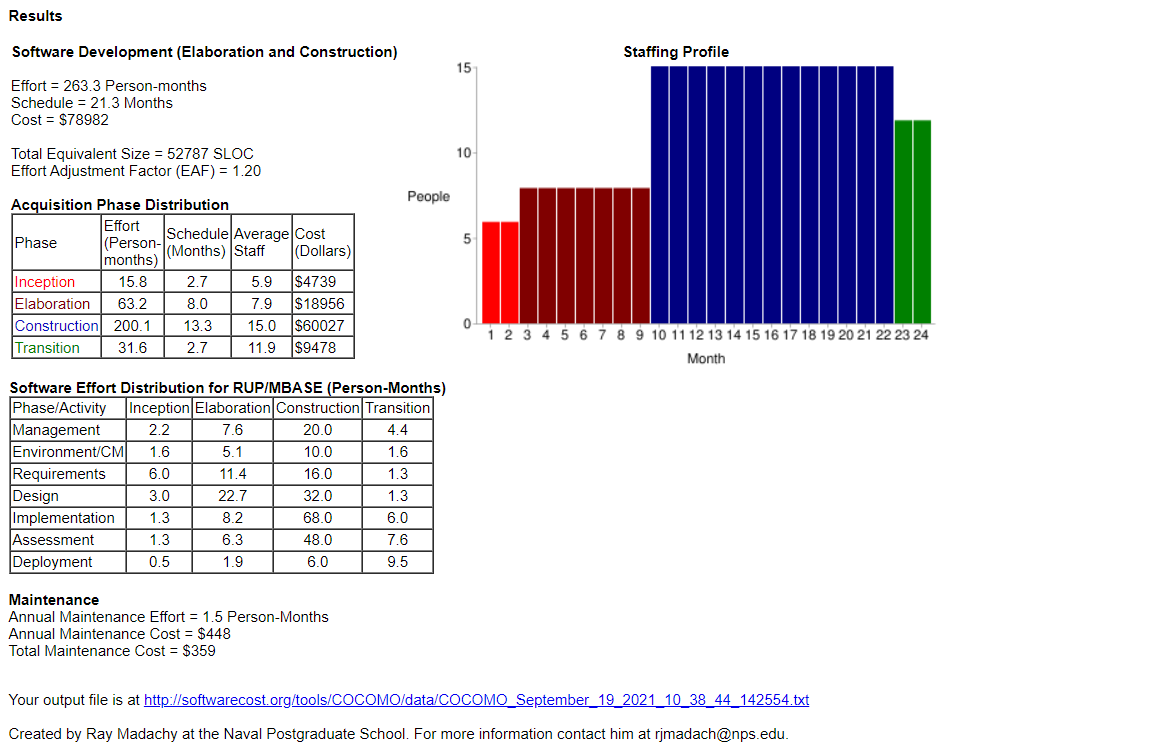


**

**

**

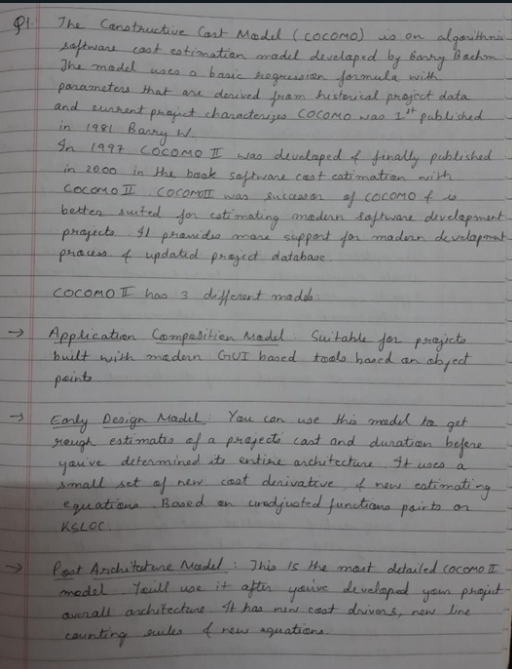




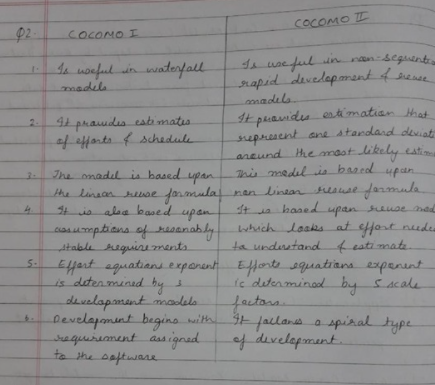
**Conclusion:** By doing this experiment we learnt the concept of COCOMO 2 model and how it can be used for project estimation.

**Post Lab Descriptive Questions**

1. Explain COCOMO II model

****

1. Compare the merits & limitations of basic COCOMO model & COCOMO II model.



1. Briefly explain the various types of for efforts & cost estimation techniques used in Software Engineering

