**Process Management Report**

February 19, 2018

Team 1 (Katharina Seitz, Daniel Lammens, Sullivan Molinek, Ricky Davidson, Ryland Wheliss)

**Introduction:**

This report describes Team 1’s management plan for its project to develop a web app for twitter trend analysis.

1. **Expected Level of Effort in Person-Months**

|  |  |
| --- | --- |
| Source | Expected LOC |
| API Client | 25 - 75 |
| Model (Processes Data) | 20 - 150 (Very uncertain) |
| Controller (Displays and calls) | 50 - 125 |
| Style (CSS and View) | 50 - 150 |
| Total\* | 145 - 500 |

\* Not including LOC auto-generated by Ruby on Rails.

Estimated Level of Effort in Person-Months (COCOMO II): 0.4 - 1.4 person-months

With Latent Dirichlet Allocation

|  |  |
| --- | --- |
| Source | Expected LOC |
| Latent Dirichlet Allocation Module | 100 - 1000 (Depends how independent we are) |
| API Client | 25 - 75 |
| Model (Processes Data) | 20 - 200 (Very uncertain) |
| Controller (Displays and calls) | 50 - 125 |
| Style (CSS and View) | 50 - 150 |
| Total\* | 245 - 1550 |

\* Not including LOC auto-generated by Ruby on Rails.

Estimated Level of Effort in Person-Months (COCOMO II): 0.6 - 4.8 person-months.

1. **Overall High-Level Schedule for the Project**

3 months. We plan to meet as a group at least once a week, most likely twice or more, to teach each other and work on the project, and have faith that we can finalize it in this time. Our project will follow the expected steps of planning and requirements which will occur simultaneously as training and skills sharing. Next, we will iterate and increment through the project. Throughout this process we can update the project requirements throughout each iteration. Finally, we will put the whole project together and then complete a quality review.

|  |  |  |  |
| --- | --- | --- | --- |
| Task | Duration (in days) | Start Date | End Date |
| Planning/Requirements | 31 | 1/28 | 2/28 |
| Training/skill sharing | 13 | 2/15 | 2/28 |
| Coding/Testing individual pieces | 42 | 3/1 | 4/11 |
| Putting it all together | 14 | 4/12 | 4/25 |
| Quality Review | 4 | 4/26 | 4/30 |

1. **Quality Plan**

Our quality goals for this project include making sure the code passes all tests, making sure the error count is below a given threshold, having reliable, easy to use, and maintainable code. To meet these goals we are going to implement different approaches to error prevention and error detection/removal. Our error prevention will consist of testing and running our code as quick as possible to make sure it is working as planned, follow preconceived standards, and communicating efficiently with team members. Our error detection and removal will consist of finding the type of error that has occurred and the source of the error and taking the appropriate steps to fix it depending on what the error is. Lastly, to make sure that our project has the best quality possible we will be implementing project wide coding standards such as documentation and good coding styles (indentation, spacing, and variable names). This will ensure that our code is usable across the board.

1. **Risk Management Plan**

|  |  |  |  |
| --- | --- | --- | --- |
| **Risk** | **Probability of Happening** | **Loss of your Project** | **Mitigation Strategy** |
| Requirements Creep | High | Medium | Clearly enumerating project requirements, checking in regularly to make sure we are sticking to the requirements |
| Lack of Accountability when sticking to schedule | Medium | High | Create a detailed schedule with benchmarks and deadlines, keep each other accountable, schedule extra meetings if necessary |
| Lack of Key Skills | Medium | Medium | workshopping skills, other training |
| Unpredictability of Davidson schedules | High | High | build in extra time, account for this |

1. **Project Monitoring Plan**

We decided to use two main measurements for our project progressing: How many lines of code our software development team has developed over time and show the success or failure of each step by looking at our prototype in progress. The goals for these measurements can be seen in the Lines of Code displayed in section 1. Each week we will make 150-155 lines of code. The amount of variance from our weekly goal will determine if we are on the right track, meeting our goals or not. We chose to make the determined lines of code to be on the lower side such that we should achieve our desired goals each week but if we are not on track we plan to make more meetings each week and switch toward a potentially more involved paired programming coding method.

1. **Detailed Scheduling**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Module** | **Task** | **Effort**  **(Days)** | **Start**  **date** | **End**  **date** | **%**  **Done** |
| - | Requirements & Planning | 31 | 1/20 | 2/20 | 100 |
| - | Design Review | 4 | 2/20 | 2/23 | 50 |
| - | Rework of Requirements | 6 | 2/23 | 2/28 | 0 |
| Update Schedule | | | | | |
| History | API Client | 10 | 3/1 | 3/13 | 0 |
| History | API Client Test | 5 | 3/14 | 3/18 | 0 |
| Update Schedule | | | | | |
| History | Model/Data Processing | 7 | 3/19 | 3/26 | 0 |
| History | Data Processing Test | 5 | 3/26 | 3/30 | 0 |
| Update Schedule | | | | | |
| History | Display and Calls | 8 | 3/31 | 4/7 | 0 |
| History | Display and Calls Test | 4 | 4/8 | 4/11 | 0 |
| Update Schedule | | | | | |
| History | Rework | 7 | 4/12 | 4/18 | 0 |
| History | Reconciliation | 7 | 4/19 | 4/25 | 0 |
| Quality | Analysis | 5 | 4/26 | 4/30 | 0 |

We will update this schedule almost weekly depending on if we are on track with our original schedule.