**PROJECT ORGANIZATION**

The team has six members: Issa Samake, Zach Danjeau, Jaquincy Nelson, Yee Wong, Brandin Jefferson, and Brian Okoye. Documentation will be handled by all team members by splitting the document evenly; the same approach has been decided for use with coding as well.

Responsibilities

Brandin Jefferson:

* Primary: User Interfaces
* Secondary: Database (File Storage), Program Documentation

Issa Samake:

* Database (Conversion), General Documentation

Zach Danjeau:

* Primary: Hardware (File scanning and conversion)
* Secondary: Database

Jaquincy Nelson

* Primary: Map construction (Map interface)
* Secondary: General Documentation

Yee Wong

* Database (Formatting/Skeleton), Program Documentation

Brian Okoye

* Team Contact, Database (Formatting/Skeleton)

These roles were chosen with each individual’s current skills in mind. If a team member had experience in a certain area, he was given a task that fit said experience. This lessens the work load by having each member only need to learn a new programming language, rather than several concepts at once. This will also allow more time for development outside of the planning stages.

**LIFECYCLE MODEL USED**

The software development lifecycle model used for this project is a mix of the waterfall and agile software development model. We used the waterfall model at the early stage of the project for the planning as well as for writing the requirements. We chose to use the waterfall model at the beginning because it was easy to understand and follow and it produced better requirements.

We used the agile model for the design, development and testing stage of the project. We used it at those stages because it was more flexible and allowed stages to overlap and run concurrently making the whole process faster. Also, it allowed adaptation to changes of requirements from the customer.

**RISK ANALYSIS**

The risk for the project include the possibilities of group members losing motivation, group members dropping out of the class, and group members deciding that they want to switch teams. This would result in staff turnover and management change risks within the project. The best way to handle these risks is for the every member in the group to try their best to complete the project. Motivation will be a combination of a finished project and a good grade. This should also keep members from wanting to switch teams.

**HARDWARE AND SOFTWARE RESOURCE REQUIREMENTS**

The hardware that will be used to carry out the development will be the computers in the CMPS lab, running on windows 7 virtual machine. Development will be also carried out on each group members individual personal laptops. Printers in the Cmps lab and other labs around U.L Lafayette campus will be used to print out documentation. We will use programming software as well as database software to implement code for the development. The Ide's will be used for interfacing and will be used alongside html script and databasing software. Ruby on the rail will be the programming language of choice as well as Mysql to help implement a database. Git will be used as a repository for the applications development. Web applications such as google doc. , and google hangout will be used as a means of communication for members within the group. Ul Lafayette email will be used to communicate with client when necessary. Facebook messenger, application will also be used for communication.

**DELIVERABLES, SCHEDULE**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Step** | **Task Name** | **Date of completion** | **Resources (Human)** | **Dependencies** |
| 1 | Define the problem | 9/19/14 | 6 |  |
| 2 | Analyze the problem | 9/25/14 | 6 | 1 |
| 3 | write the requirements documentation | 9/30/14 | 4 | 2 |
| 4 | Determine Technologies and tools to be used | 9/30/14 | 6 | 1 |
| 5 | Start training with new technologies and tools | 10/8/14 | 6 | 4 |
| 6 | Write project plan | 9/30/14 | 2 | 2 |
| 7 | Design solution | 10/9/14 | 4 | 3 |
| 8 | Write architecture Documentation | 10/9/14 | 2 | 7 |
| 9 | Development phase | 11/10/14 | 5 | 8 |
| 10 | Write Test Plan | 11/15/14 | 2 | 9 |
| 11 | Tesing and Debugging phase | 11/25/14 | 2 | 10 |
| 12 | Deployment | 11/26/14 | 6 | 11 |
| 13 | Final Project Report | 11/27/14 | 6 | 10 |



**MONITORING, REPORTING, AND CONTROLLING MECHANISMS**



**PROFESSIONAL STANDARDS**

**Scholastics Dishonestly:**

Team members programming material should be completely of their own work. On the first occurrence, notify the instructor of the problem. A meeting will be set up to evaluate the situation and resolve the problem. On the second incident, the team will notify the instructor of the problem and a meeting will be setup to resolve the current issue. On the third occurrence, the instructor will be notified and a meeting will be held; the team may have an option of removing the offending member.

**Meeting Schedules:**

Each team member should attend a majority of team meetings. In case of a valid reason, a team member/s may setup another meeting to review any prior meeting scheduled. On the second incident, the team will notify the instructor of the problem and a meeting will be setup to resolve the current issue. On the third occurrence, the instructor will be notified and a meeting will be held; the team may have an option of removing the offending member.

**Quality Expectations:**

Each member should attempt to execute a higher quality of work based on the task at hand.

On the first occurrence of a member lacking in work quality, a team meeting will be held and resolved by team discussion. On the second incident, the team will notify the instructor of the problem and a meeting will be setup to resolve the current issue. On the third occurrence, the instructor will be notified and a meeting will be held; the team may have an option of removing the offending member.

**EVIDENCE OF CONFIGURATION MANAGEMENT**

Git Repository: https://github.com/brandinjefferson/CMPS453-Docs

