**Software Project Management Plan**

**Team 3**

December 12, 2019

**Team Members**

Enrico Marchi

Elia Gaole

Giulio Federico Grippi

Jennifer Emiliani

Matteo Alexandru Bulai

Document Control

**Change History**

|  |  |  |
| --- | --- | --- |
| **Revision** | **Change Date** | **Description of changes** |
| V1.0 | 12/12/19 | Initial release |
|  |  |  |

**Document Storage**

This document is stored in the docs’s repository at:

<https://github.com/enricomarchidev/Potatolk.git>

**Document Owner**

Gaole Elia is responsible for developing and maintaining this document.

**Table of Contents**

[**1 OVERVIEW 5**](#_heading=h.gjdgxs)

[1.1 Purpose and Scope 5](#_heading=h.30j0zll)

[1.2 Goals and Objectives 5](#_heading=h.1fob9te)

[1.3 Project Deliverables 6](#_heading=h.3znysh7)

[1.4 Assumptions and Constraints 6](#_heading=h.2et92p0)

[*1.4.1 Assumptions 6*](#_heading=h.tyjcwt)

[*1.4.2 Constraints 6*](#_heading=h.3dy6vkm)

[1.5 Schedule and Budget Summary 6](#_heading=h.1t3h5sf)

[*1.5.1 Cost Estimate 6*](#_heading=h.4d34og8)

[*1.5.2 Schedule Summary 7*](#_heading=h.2s8eyo1)

[1.6 Success Criteria 8](#_heading=h.17dp8vu)

[1.7 Definitions 8](#_heading=h.3rdcrjn)

[**2 STARTUP PLAN 9**](#_heading=h.26in1rg)

[2.1 Team Organization 9](#_heading=h.lnxbz9)

[2.2 Project Communications 9](#_heading=h.35nkun2)

[2.3 Technical Process 10](#_heading=h.1ksv4uv)

[2.4 Tools 10](#_heading=h.44sinio)

[**3 WORK PLAN 10**](#_heading=h.2jxsxqh)

[3.1 Resource Estimate 10](#_heading=h.z337ya)

[3.2 Release Plan 10](#_heading=h.3j2qqm3)

[*3.2.1 Plan By Feature 10*](#_heading=h.1y810tw)

[*3.2.2 Flow Chart 11*](#_heading=h.4i7ojhp)

[3.3 Iteration Plans 11](#_heading=h.2xcytpi)

[*3.3.1 First Iteration 11*](#_heading=h.1ci93xb)

[**4 CONTROL PLAN 11**](#_heading=h.3as4poj)

[4.1 Monitoring and Control 11](#_heading=h.1pxezwc)

[4.2 Configuration Management Plan 12](#_heading=h.2p2csry)

[**5 SUPPORTING PROCESS PLANS 13**](#_heading=h.3o7alnk)

[5.1 Risk Management Plan 13](#_heading=h.23ckvvd)

[5.2 Test Plan 13](#_heading=h.32hioqz)

[5.3 Product Acceptance Plan](#_heading=h.1hmsyys)  13

1. **Overview**
   1. **Purpose and Scope**

Team 3 became interested in creating a client application capable of communicating with a server application developed by another team. This application, which the team wanted to call **Potatolk**, is basically a chat.

It is therefore able to exchange messages with other client applications, created by other teams, through communication with the server.

The application will offer the possibility, once logged in omitting the topic, to communicate in a general room, then an area used by all, from the general room you can then move on to communication with individual users.

If, however, always in the login phase, the topic will not be omitted, you will move to a sort of private room in which you can communicate simultaneously with all users who have entered the same topic.

The user interface will be intuitive, simple to navigate and beautiful to the eye.

* 1. **Goals and Objectives**

The general objective is to make clients created by different teams communicate with each other.

Objectives of the project:

1. Create an application that works as expected and looks good.
2. Information on how java swing works.

Project goals:

1. Create a mobile interface with java swing
2. Create an application that functions in a simple and intuitive way.

* 1. **Project Deliverables**

|  |  |
| --- | --- |
| **Date** | **Deliverable** |
| 01/11/19 | Started Project |
| 08/11/19 | initiated a first idea of graphic |
| 15/11/19 | Worked on the registration packages |
| 22/11/19 | merge GUI to the methods created |
| 29/11/19 | Worked on message exchange packages |
| 06/12/19 | Updated the graphic components |
| 12/12/19 | Iteration #1 Plan |
| 12/12/19 | Iteration #1 Complete |
| 12/12/19 | Product Released |

* 1. **Assumptions and Constraints**
     1. ***Assumptions***

1. The application are available and function when the server is active.
   * 1. ***Constraints***
2. The application need specific libraries.

* 1. **Schedule and Budget Summary**
     1. ***Cost Estimate***

1 project manager at 4 hours per week for 14 weeks 56 hours \* $50/hr = $2800

1 requirements engineer at 4 hours per week for 14 weeks 56 hours \* $40/hr = $2240

2 software engineers at 4 hours per week each for 14 weeks 112 hours \* $40/hr = $4480

224 hours total, $9520 total, avg, $42.50 per hour

* + 1. ***Schedule Summary***

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Step | Iteration | 1 | | | | | | | | 2 | | | 3 | | |
| Week | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 |
| End Date | 09-10 | 09-17 | 09-24 | 10-01 | 10-08 | 10-15 | 10-22 | 10-29 | 11-05 | 11-12 | 11-19 | 11-26 | 12-03 | 12-10 |
| Design Architecture | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Specify/Evaluate Requirements | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Iteration 1 Development | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Iteration 1 Analysis | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Iteration 2 Design | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Iteration 2 Development | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Iteration 2 Analysis | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Iteration 3 Design | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Iteration 3 Development | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Iteration 3 Analysis | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

* 1. **Success Criteria**

A functional, easy-to-use prototype that allows users to easily send and receive messages.

* 1. **Definitions**

|  |  |
| --- | --- |
| **Term** | **Definition** |
| **Actor** | user or other software system that receives value from a user case. |
| **Baselined** | the work product has undergone a formal review and can only be changed through the prescribed change control procedures |
| **Client or Customer** | the person or organization for which this Roo Balance application is being built. |
| **Developer** | the person or organization developing the system, also sometimes called the supplier. |
| **Project** | activities that will lead to the production of the Roo Balance application. |
| **Roo Balance Application** | the product that is being described here; the software system specified in this document. |
| **Scenario** | one path through a user case |
| **Stakeholder** | anyone with an interest in the project and its outcomes. This includes clients, customers, users, developers, testers, managers and executives. |
| **User** | the person or persons who will actually interact with the Roo Balance application. |
| **Use case** | describes a goal-oriented interaction between the system and an actor. A use case may define several variants called scenarios that result in different paths through the use case and usually different outcomes. |

1. **Startup Plan**
   1. **Team Organization**

|  |  |  |
| --- | --- | --- |
| **Role** | **Actor(s)** | **Responsibility** |
| Project Manager | Marchi | Call team meetings, coordinate communications within group, coordinate communications outside group, break out tasks, assign them to teammates |
| Developer | Gaole, Grippi | Develop software based on requirement and architect specifications |
| Programmer | Marchi | Program to requirement and architect specifications |
| Tester | Marchi, Gaole,  Bulai | Write test cases, perform unit testing of test cases against incremental release of code, perform integrated testing of test cases against incremental release of code, report issues |
| Architect | Marchi, Gaole | Specify overall internal workings of application |
| Requirement Engineer | Emiliani | Outline and document project dependencies and requirements. This includes internal and external dependencies. |

* 1. **Project Communications**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Event** | **Information** | **Audience** | **Format** | **Frequency** |
| Team Meeting | Task status: completed since last meeting & planned for next;  obstacles encountered; change requests in process | All team members | Informal meetings following class; Formal meetings as needed; E-mail status updates & problems as they occur | As needed |
| Project Status Report | Review finished items, status of prototype; review any problems, schedule slippage, programming issues | All team members, customer | E-mail with information or In-person as customer sees fit | Iteration Closeout |

* 1. **Technical Process**

An iterative and incremental development process is planned. The first iteration will focus on the basic functionality of the application. Subsequent iterations will be based on improving these features.

* 1. **Tools**
* Programming & Markup Languages – Java Swing
* Operating System – Windows
* Version Control – all work products will be stored in a repository
* Development Tools – Netbeans

1. **Work Plan**
   1. **Resource Estimate**

Detailed resource estimates are available in the linked file [Team 3 Estimated Effort](https://github.com/enricomarchidev/Potatolk/graphs/contributors).

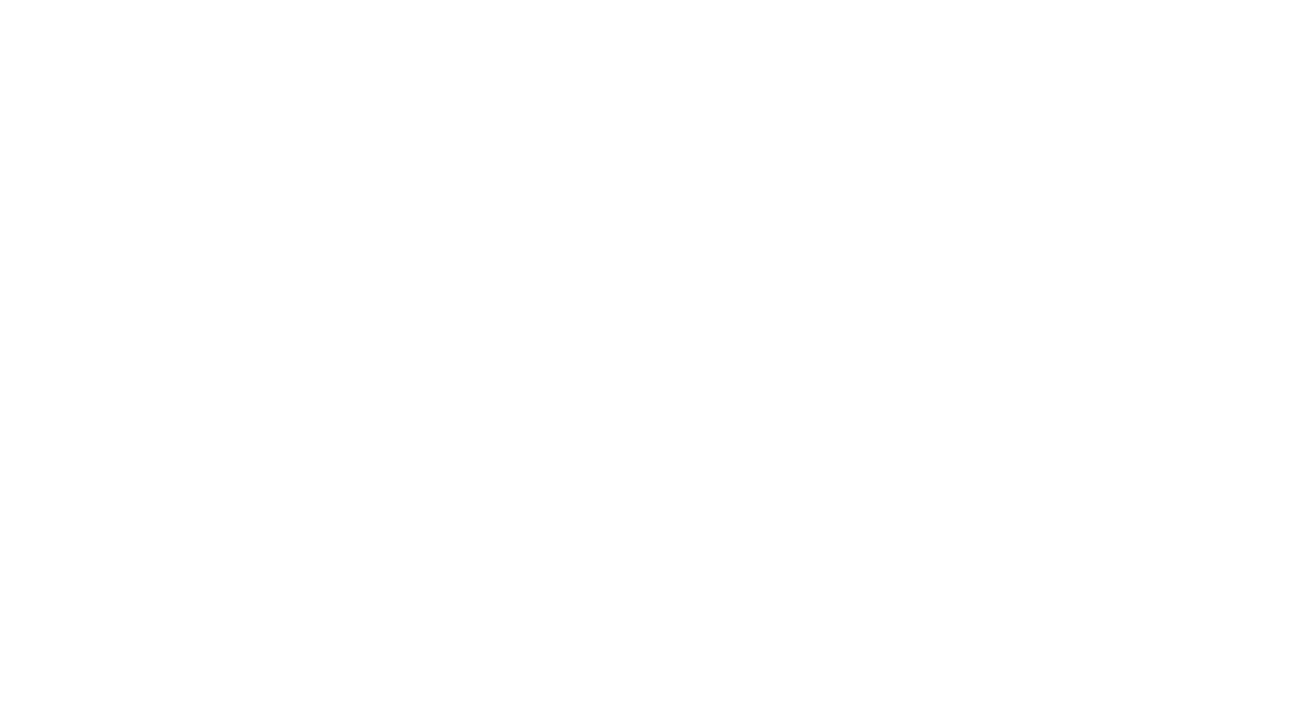
* 1. **Release Plan**
     1. ***Plan By Feature***

**Iteration #1 13/12**

**Summary:** Demonstrate fundamental architecture.

|  |  |  |
| --- | --- | --- |
| *Features / Deliverables* | **Estimated Effort** | **Actual Effort** |
| Architecture / framework design | 20 |  |
| Authentication / Credential management | 50 |  |

* + 1. ***Flow Chart***



* 1. **Iteration Plans**

A detailed iteration plan will not be provided for Iteration 1.

* + 1. ***First Iteration***

The client communicates correctly with the server and the graphic part is perfectly merged with the various methods created

1. **Control Plan**
   1. **Monitoring and Control**

The following list of dates includes formal reviews outside of the Communication Plan. Milestones are included to reference where the project is scheduled to stand as these reviews occur:

|  |  |
| --- | --- |
| **Date** | **Review / Milestone** |
| *08/11/2019* | *Milestone: Initial idea for graphic* |
| 08/11/2019 | Generated three different windows (login, logout & chat) |
| 08/11/2019 | Analyzed the colors |
| *15/11/2019* | *Milestone: Started to work with packages* |
| *12/12/2019* | *Milestone: Iteration #1 Complete* |
| *12/12/2019* | *Milestone: Product Released* |
| 12/13/2019 | Final Presentations |

* 1. **Configuration Management Plan**

The following procedure is to be used when making changes to all baselined work products:

1. All project work products will be stored in a Github repository running on a central server.
2. All baselined documents will have a Document Control section with a change history to track initialization and subsequent changes.
3. All project work products (documents, source code, test cases, program data, test data, etc) will be stored in the repository but not all will be under change control. Only the system requirements, project plan and source code will be baselined and under configuration control.
4. Items that are subject to change control will be considered baselined after a group review at the end of the initial document creation.
5. The change control procedure once a product is baselined is:
   1. anyone wanting to make a change to a baselined item sends an email to the rest of the team and project sponsor (i.e. Professor Burris) describing the change, reason for the change, expected schedule impact, and timeline for integrating the change.
   2. if no one responds to the group within 2 days with a reason for why the change request shouldn't be permitted, it will be considered accepted and the person proposing the change may proceed with the change.
   3. if anyone does object to the change, the reason for objecting will be discussed at a meeting where everyone is invited to attend and voice their opinion. At the end of the meeting a democratic vote will be held to decide whether or not the change should be allowed.
   4. if a change takes place, the initiator must collaborate with the project manager to update the schedule

1. **Supporting Process Plans**
   1. **Risk Management Plan**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Rank** | **Risk** | **Probability of Loss** | **Size of Loss** | **Risk Exposure** | **Response** |
| 1 | Schedule / time line delivery | Likely | Major | High | Mitigate: Stick to the schedule. |
| 2 | Learning curve for new tools and technologies longer than expected | Unlikely | Moderate | Moderate | Buy Information: Begin working on a basic prototype early to test out fundamental programming concepts & knowledge |

* 1. **Test Plan**

The test plan defines the items that will be tested, methods for testing, and a schedule detailing the tasks, owners, and timeline.

The test plan will not be available.

* 1. **Product Acceptance Plan**

At the end of the iteration, the prototype created will be tested with the other clients to verify the operation of the application with the others. a bogus server can be created to test client functionality.