Tampere University Unit of Computing Sciences

TIEA4 Project Work (City centre campus)

TIETS19 Software Project Management Practice (City centre campus)

Software component

Software Component manager

Project plan

|  |
| --- |
| 424990, Jyry Uitto |
| 431258, Petri Nieminen  427770 Jani Aakio  439656, Kaisa Sormunen K427803 Pekka Oinas  K427534, Sanni Onkalo  K439136, Denis Kole |

Version history

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Author | Description |
| 0.1 | 29.1.2021 | Project team | First draft |
| 0.2 | 5.2.2021 | Jyry Uitto | Detail additions |
| 1.0 | 16.2.2021 | Jani Aakio | Cleaning up |
| 1.1 | 24.2.2021 | Jyry Uitto | Risk mitigation details |
| 1.2 | 20.5 | Jyry Uitto | Updated details |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Contents

1 Introduction 4

1.1 Purpose and scope of project 4

1.2 Product and environment 4

1.3 Customer's current system and other similar systems 4

1.4 Project constraints 4

1.5 Definitions, abbreviations and acronyms 4

2 Project organisation 5

2.1 Group members 5

2.2 Customer 5

2.3 Related organisations 5

3 Project goals and ending/termination 5

3.1 High level goals of the project group 5

3.2 High level goals of the customer 5

3.3 High level goals and deliverable(s) of the project 5

3.4 Quitting (termination) criteria of the project 6

3.5 Ending criteria of the project 6

4 Project and Process management 6

4.1 Methods and tools 6

4.2 Planned meetings 6

4.3 Learning and study plan 7

5 Project iterations and timing 7

5.1 Iterations (sprints) 7

5.1.1 Iteration 0 8

5.1.2 Iteration 1 (first implementation sprint) 8

5.1.3 Iteration x 8

5.1.4 Iteration z (quality assurance) 8

6 Requirements 9

6.1 Functional requirements (main goals) 9

6.2 Non-functional requirements goals 9

6.2.1 Usability goals 9

6.2.2 Performance goals 9

6.2.3 Reliability goals 9

6.2.4 Security goals 9

6.3 User interface requirements (main goals) 9

7 Risk management 9

7.1 Risk list 10

7.2 Risk monitoring 10

8 References 11

9 Open issues 11

10 Ideas for further development 11

APPENDIX A […Z] 11

# Introduction

## Purpose and scope of project

This document is a framework for the product Software Component manager. This document will provide all necessary details and references to documents made about this product and aid with the completion of the project. Furthermore, it shall help with project management and will assure that everything is done that should be done. This document will be updated until the end of the project.

## Product and environment

Software component manager.

Software made is at core a database that holds information about software components.

Further possible features provided are:

* The ability to check if proper software components are installed for a specific software.
* Check required software components for a specific product and version of them.
* Web user interface
* User authentication and security
* Understandable form of input and output methods and data.

This product is made for Accenture for all intents and purposes.

## Customer's current system and other similar systems

The software is to be integrated into the CI pipeline. Although it is going to be integrated into a system, it should be a stand-alone product and has no dependencies on the customers system.

Nexus repository and perforce are commercial products that are very similar to the software being developed. These products are out-of-the-box solutions that may not fit all requirements of the software.

## Project constraints

Constraints are very few. Some preferences on technologies/languages have raised such as Scala, Angular, Postgres, REST, openapi. The team has been given the permission to use other overriding technologies, but the customer could provide support with these aforementioned technologies.

Also, security is a large non-functional requirement which has been brought up by the customer and will be dealt with accordingly by the development team.

Accenture has all rights to this product.

## Definitions, abbreviations and acronyms

UI User interface.

REST Representational State Transfer

# Project organisation

## Group members

Jani Aakio, 0408412601, [Jani.aakio@tuni.fi](mailto:Jani.aakio@tuni.fi), Project manager

Jyry Uitto, 0400602108, [Jyry.uitto@tuni.fi](mailto:Uitto,0400602108,Jyry.uitto@tuni.fi), Project manager

## Customer

Accenture

Juha Tuominen, 0405040774, juha.t.tuominen@accenture.com

## Related organisations

Tampere University.

Pekka Mäkiaho, 0505566266, Project supervisor

Timo Poranen, Course coordinator

# Project goals and ending/termination

## High level goals of the project group

* Making of a working piece of software that does what it should.
* Staying in schedule
* Customer satisfaction
* Every project member learns something meaningful.

## High level goals of the customer

Priority of the (technical/business) goals: primary goal, secondary goal, etc.

Customer goals can be for example:

* Product should do its functions reliably.
* Show required software components.
* Show missing/changed software components.
* Is a secure piece of software.

## High level goals and deliverable(s) of the project

* Web-UI
* Database
* Authentication and security
* Data input
  + Checking a software and its components
  + Inputting new data
* Data output
  + Default format is JSON
* Possibility of deployment
* Project end-report and test-report

## Quitting (termination) criteria of the project

Definition of situations when the project is terminated:

* Customer terminates the project.
* Available resources end.

## Ending criteria of the project

Ending criteria:

* Deadline at 14.5.2021.
* The project is finished.
* Working product.
* All available working hours are used.

# Project and Process management

## Methods and tools

* Version control system (github)
* Kanban board
* Documentation for customer:
* Solid code and no useless comments.
* Documentation for Tuni:
* Project document and MMT reports.
* Change management
* Risk management.
* Project end-document
* Development method is scrum-BUT,

At project courses MMT tool is used for weekly reporting and working hours logging.

## Planned meetings

Team meets weekly on Wednesday 16:00 in a Teams meeting. Purpose is to update schedule between members.

## Learning and study plan

* Docker, tutorialized by Jani
* Other tools that might be used overall.
* For some:
* Scala
* Angular
* Postgres
* Rest, tutorialized by Jani
* Openapi3, tutorialized by Jani
* WSO2
* OWasp

# Project iterations and timing

## Iterations (sprints)

Details are at Backlogs (group’s Kanban tool).

8 Sprints with average length of 2 weeks.

**Table 5.1. Project schedule.**

|  |  |
| --- | --- |
| Sprint no | Schedule (start/end date) High level content. |
| 1 | 22.1 - 9.2 Project setup |
| 2 | 9.2 - 23.2 Hello world |
| 3 | 23.2 - 9.3 Minimum viable |
| 4 | 9.3 - 23.3 (Minimum viable?) + initial security |
| 5 | 23.3 - 6.4 |
| 6 | 6.4 - 20.4 |
| 7 | 20.4 - 4.5 |
| 8 | 4.5 - 14.5 Polishing |
| 9 | 14.5 - 26.5 Polishing (Additional iteration) |

### Iteration 1

* Tools and work environment setup.
* Work distribution between members.

### Iteration 2 (first implementation sprint)

* Implement “hello world” for future containers
* Get familiar with docker
* Implement simple http calls for containers

### Iteration 3

* Have a “minium viable product” available
  + Basic data insertion to db
  + Basic data retrieval from db
  + Simple UI

### Iteration 4

* Have a “minium viable product” available
  + Security initialization
  + Basic functionality through UI

### Iteration 5

* Add components and utilities to existing product
  + Security initialization
  + Add functionality to UI
    - Insert
    - Delete
    - Queries
    - Compare
  + Initialize testing

### Iteration 6

* Add components and utilities to existing product
  + Security initialization
  + Add functionality to UI
    - Insert
    - Delete
    - Queries
    - Compare
  + Add tests

### Iteration 7

* Add components and utilities to existing product
  + Security initialization
  + Add functionality to UI
    - Insert
    - Delete
    - Queries
    - Compare
  + Make existing tests pass.

### Iteration 8 (quality assurance)

* Add components and utilities to existing product
  + Security Components removed
  + Add functionality to UI
    - Insert
    - Delete
    - Queries
    - Compare
  + Add tests

### Iteration 9 (quality assurance)

* Add final components and utilities to existing product
  + Test product for bugs.

# Requirements

## Functional requirements (main goals)

* Software should have Web-UI
* User authentication
* Input validation
* Database
* Information to save
* Software, version number, dependencies
* Comparing of software version dependencies
* Data-input
* Data-output
* Program should determine required software to run a program

## Non-functional requirements goals

### Usability goals

* Data can be managed through data input and output channels only. (No necessary UI)

### Security goals

* Minimum of basic security features (WSO2)
* Possibility of real user security (database) and possible integration with other systems

# Risk list

**Table 7.1. Project risks.**

| Risk ID | Explanation, severity/impact, probability, size/importance | Impact/Severity | Probability |
| --- | --- | --- | --- |
| 1 | Customer drops the project | Very High | Very Low |
| 2 | Members lack motivation/time to do the project | High | Low |
| 3 | Sickness of team members / key individuals | Low | Medium |
| 4 | Lack of required skills | Medium | High |
| 5 | Hardware problems | Medium | Low |
| 6 | Not well-defined requirements | High | Very Low |
| 7 | Software integration issues | High | Very Low |

## Risk mitigation

1 Customer drops the project:

* Mitigate risk by:
  + Since probability is very low no actions are done until risk occurs.
* In case the risk occurs:
  + Continue project as is.
  + Take contact to Tiea course personnel and define what must be done for the project to achieve its goals.

2 Members lack motivation/time to do the project

* Mitigate risk by:
  + Try not to overachieve with project goals.
  + Take less goals at once and achieve them and only afterwards take new requirements.
  + Keep eyes on team member work hours, try to motivate people if some seem to be lacking behind.
  + Keep contact up with members that are working on critical parts.
* In case the risk occurs:
  + Delegate work to other members of the team.
  + Before taking new requirements, make sure that current ones can be accomplished.

3 Sickness of team members / key individuals

* Mitigate risk by:
  + Keep contact up with members that are working on critical parts.
* In case the risk occurs:
  + Delegate work to other members of the team.
  + Before taking new requirements, make sure that current ones can be accomplished.

4 Lack of required skills

* Mitigate risk by:
  + Delegate some areas that each team member should learn.
  + Teach other members the key things that are required.
* In case the risk occurs:
  + Ask help from team members.
  + Ask help from Accenture.

5 Hardware problems

* Mitigate risk by:
  + Maintain good version control.
  + A way to make sure that all progress cannot be lost under any circumstance.
* In case the risk occurs:
  + -

6 Not well-defined requirements

* Mitigate risk by:
  + Keep client in development process.
  + Show completed requirements and make sure that they are what they were supposed to be.
  + If questions arise what this requirement means, ask them.
* In case the risk occurs:
  + Build the more well-defined features.
  + Add missed requirements as features if they are usable.

## Risk monitoring

Risk management is done at MMT tool when doing weekly reports by the person doing such report.

# References

Harold Kerzner: Project management: a systems approach to planning, scheduling, and Controlling, 2013, Wiley.

Scrum Guide, 2017.

# Open issues