

|  |
| --- |
| Shore Lines  2018 |
|  |
| May 31  Binary  Authored by: Kristófer, Jacob, Kasper and Skomantas |

Table of Content

Introduction 3

Background 1.1 3

Problem Definition 1.2 4

Product Vision 1.3

Strategic Analysis 1.4 #

PreGame #

Project Organization 2.1 #

Overall Project Schedule 2.2 #

Initial Product Backlog 2.3 #

Architecture 2.4 #

Sprint 1 #

Sprint Planning 3.1 #

Daily Meetings 3.2 #

GUI 3.3 #

**Data Model 3.4** #

**Implementation 3.5** #

**Sprint Review 3.6 #**

**Sprint Retrospective 3.7** #

Sprint 2 #

Sprint Planning 4.1 #

Daily Meetings 4.2 #

GUI 4.3 #

**Data Model 4.4** #

**Implementation 4.5** #

**Sprint Review 4.6 #**

**Sprint Retrospective 4.7** #

# Introduction

|  |
| --- |
| **Background 1.1** This project consisted of using SCRUM, a framework that allows a team such as ours to effectively and productively collaborate on a complex system such as the assignment we were given by EASV teachers. It allowed us to organize our time and helped every member to evaluate the amount of time each operation would take. It gave the members a better insight in to a more professional environment with a good team structure, time management and overview of the amount of work each member has contributed. |
| *“It gave the members a better insight in to a more professional environment with a good team structure”* |
| The company SHORELINE cooperated with the teachers and helped forming the standard of what the company expected from the project, participated in SCRUM meeting as well to give feedback and ask question regarding the program features/GUI that was presented during the meetings.  The assignment we were given was to create a program that would convert files such as XML, XLSX and CSV in to JSON files, so it could be used with SHORELINE web platform/system, which reads only JSON formatted files. The program had to have a way for the user to customize the configuration on how the data would be converted, including checking if data that is being imported is valid and a form of traceability to track what each user did, when the user did it and errors that would display what went wrong with the conversion.  Binary group agreed on using a login feature where the user would login with given username and password, this way the program could trace each user, what they did and when. All this information would be displayed in the log window, where users can see the error if there was any. The main functionality of the program is to convert file types XLSX and CSV in to JSON format, allow users to interact with the program simultaneously to the conversion, pause, stop and resume the task. The program was to be written using JavaFX and SQL database to store relevant data. **Problem Definition 1.2** *“Shoreline conducts a wide range of simulations. A lot of the data for the simulations are done based on data coming from other platforms/systems”.*    Shoreline works with various companies around the world, they simulate data, construction cost, consulting services and overall analysis of lifecycle for the wind turbines. This helps the companies to avoid additional cost and utilize energy more efficiently. They receive data from these companies, but the data format varies between companies, configuring this data manually is very time-consuming process.  “*Shoreline needs a tool that can migrate/convert data from a range of platforms, to their own web-platform*”. This will save a lot of time and resources as opposed to having to manually go through all the data and import in to their own system.  **Figure 1** *Problem Definition explained with a solution.* |
|  |

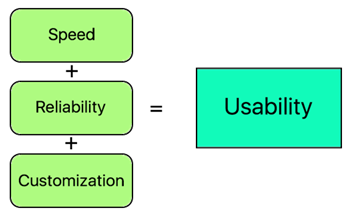
Product Vision 1.3

Our plan from the beginning was to create an exceedingly simple project, so that it would be easy for companies to train people in it, and therefore the fewest number of hours would be wasted should they decide to move to this program.

It had to be a fast-acting program, where you could start converting files right away. The product also had to do the converting itself at a fast rate, as companies will often have thousands of files. Usability, reliability and speed, these were the primary goals for our program.

Another big goal we had, was that users should be able to customize their data so that it would fit their purposes perfectly, and should these purposes change, then they could easily edit how the custom data looked. This was a very high priority for us, as we felt this is where the program could stand out from others. We wanted the program to feel like it was designed specifically for that one company, even though it may be used by many different.

Figure



Strategic Analysis 1.4

There were a lot of discussions among team members about what our strategy would be with this product, what we all agreed upon was that it had to be simple to use and that it had to be a very useful program with practical options. It didn’t need to be extremely advanced with a million options, it just needed to work and be effective. The most important feature of the program was to convert Excel files to Json.

Binary created the program from this point of view, and had some tests done with a few test persons. To see how difficult is to convert the data. From the testing result we decided that the program was good enough. So, we decided to add more options like custom data and log view.

|  |  |
| --- | --- |
| Risk | Risk Management |
| Human |  |
| Illness | The control of the risk: First the person should inform group members that he is staying at home because of illness. Secondly the group should decide. Is the person continuing work from home? Or the group should divide his part of work and do it all together |
| Injury | The control of the risk: If the injury does not cause serious risk for the persons health and he can fully use the computer than he should continue his work from home. |
| Technical |  |
| Not working electronic | The control of the risk: Our work and DATA are shared in the group by cloud, it means that any electrical difficulties will not affect us. Unless we are going to have some problems with cloud servers (Internet). |
| Project |  |
| Not managing to do tasks on time | The control of the risk: Our time on every task is controlled by SCRUM. If any of the task is on late we can see that from far away. We can plan how to avoid that late. |
| Project quality problem | The control of the risk: We are responsible for the quality that's why we need to control our work and check with every new feature in the program. |

Green – Low Risk

Yellow – Medium Risk

Red – High Risk

# Pregame

# Project Organization 2.1

Binary SCRUM team consists of four members, Jacob, Kristófer, Kasper and Skomantas. Each member contributing in their own way to the assignment given.

A task varies in difficulty, therefor assigning them requires careful planning, which may lead to two members working on the same user story with different task. The team had previous experience working on a project together and it worked well, the roles stayed relatively the same with some improvements. Jacob was the representative of the product, showing the functionalities to the clients and discussing the features, discussing with team members the product vision. SCRUM master was Kristófer, who made sure that the product vision was followed and set up the standup meetings, enabling communication between team members and working with the product owner regarding product vision and role assignments. Product owner and SCRUM master had twofold roles, being part of the team was the secondary role.

Figure Binary Team Setup

A close up of a sign

Description generated with high confidence

# Overall Project Schedule 2.2

# *The Binary SCRUM team Problem Definition was done in one day, having the product owner and SCRUM master making sure the vision would be followed throughout the project, never straying too far from the vision with exceptions being post SCRUM meetings with the client, giving feedbacks to edit features or add them for improved usability. Product Vision was decided on the first day, discussed during Standup meetings which we had two times per week, usually at the beginning of the week and the end.*

|  |  |  |
| --- | --- | --- |
| SCHEDULE |  |  |
| Planning |  |  |
| Name | Time table - Start | Time table - End |
| Problem Definition | 2018-04-23 | 2018-04-25 |
| Product Vision | 2018-04-23 | 2018-04-25 |
| Strategic Analysis | 2018-04-23 | 2018-04-24 |
| Project Organization | 2018-04-23 | 2018-04-23 |
|  |  |  |
| Design |  |  |
| Name | Time table - Start | Time table - End |
| Login feature | 2018-04-24 | 2018-04-26 |
| Log feature, track users | 2018-04-27 | 2018-05-05 |
| Conversion feature I/O | 2018-04-24 | 2018-05-23 |
| Create user feature | 2018-04-30 | 2018-05-01 |
| GUI fixes and various small tasks | 2018-04-24 | 2018-05-31 |
|  |  |  |
| Report |  |  |
| Name | Time table - Start | Time table - End |
| Writing the Report | 2018-05-09 | 2018-05-31 |

# Initial Product Backlog 2.3

Figure Initial Backlog created April 23d, Login screen looks relatively the same while the main import window and export have improved significantly, the Log view is the same as well. Made by Product Owner and SCRUM Master.

# A close up of text on a white background Description generated with high confidence

# The Binary group got together after the first meeting with the client and analyzed the problem definition and worked on Product vision. Started working on Backlog on sheet of paper, where ideas and vision would come together. Simplicity was the key, creating a user-friendly program without too many distractions, enabling the user to open the system and start the task with relative ease. The very next day the group got together again to create a Backlog Items on SCRUMWISE.COM, time estimate was made by using pokerplanning.com and focus percentage which stayed the same throughout the project. Our estimate was not too far off and we managed to stay in the timeframe and complete functional features before the first SCRUM meeting.

Figure Binary first initial backlog, created on April 24th, second Standup Meeting.

# A screenshot of a cell phone Description generated with very high confidence

# Architecture 2.4

As we have been taught throughout the 1st and 2nd semester, we decided to go with the Three-tier architecture. We felt more at home with this type of architecture, and felt it fit the client-server application that we were going to build. We felt it was a perfect fit since the program had to be developed for a lot of users, that was all going to be fed into our set of databases.

We used JavaFX implementing the Model-View-Controller (MVC) pattern, so that we could easily take use of the FXML documents we’d be creating in Scenebuilder.

As we needed to be able to de-couple layers, we ended up using the Façade pattern, as this is what we felt most comfortable with as we had previous experience with it.

*Figure 5 This is our implementation of the MVC Pattern*

A screenshot of a cell phone

Description generated with high confidence

# 