

Project Report

Reviewing Decision

Support System

Group Mjólnir:

Petroula Theodoridou

Calle Persson

Simon Lobo

Natalie Davidsson

Yurou Zhao

Peili Ge

Table of Contents

Project Overview.....	4
Purpose of this document.....	4
Acknowledgements.....	4
Introduction.....	5
Background Information.....	5
Definitions, Acronyms and Abbreviations.....	5
The Project.....	6
System Definition.....	6
The Problem Domain.....	7
Clusters.....	7
Detailed Project Structure.....	8
Application Domain.....	9
Functionalities.....	9
Structure And Organisation.....	13
Usefulness and Feasibility.....	13
Software Process Strategy.....	13
Team Organisation.....	14
Detailed Individual Work.....	15
Project Plan.....	15
Coding.....	16
Project Report.....	17
Schedule.....	18
Testing and Validation.....	19
Strengths vs Weaknesses.....	19
Evaluation.....	20
Changes to the Final Product.....	21
Requirements.....	21
Requirements Summary.....	21
Major Challenges and Solutions.....	22
Decision Making.....	22

Choosing Design.....	22
Database.....	23
Editing the Interface.....	23
Combining Code.....	23
Prioritizing.....	24
Repeated Code.....	24
Conclusion and Future Work.....	25
Conclusion.....	25
Future Work.....	26
References.....	27
System User Manual.....	28

Project Overview

As a six-member group of first year students enrolled in *Software Engineering and Management, Bachelor's Programme* at the University of Gothenburg we were assigned the task of developing a Decision Support System (DSS) of our choice. The DSS in reference is a Java application that handles reviews for books, movies and games.

Purpose of this Document

The main purpose of this document is to provide teachers of the *Software Engineering and Management, Bachelor's programme*, the program's supervisors, the Mjolnir group and potential customers with the final result of this project. This report essentially aims at providing sufficient details explaining our complete product.

Acknowledgements

In this section the Mjolnir group consisting of Calle Persson, Natalie Davidsson, Simon Lobo, Petroula Theodoridou, Peili Ge and Yurou Zhao would like to give their appreciation and gratitude for the help provided by our supervisor: Thor Salehi. Thor has kept us on track and given us feedback, criticism and useful advice where it was greatly needed. Since this was the first time most members in the group were introduced to a programming project we did our best to turn the given feedback into a driving force that led us throughout the project.

Introduction

With our Product Betacritic our aim was to allow users to find different medias they enjoy, without the hassle of extensive research. To do this we provide the users with the most popular games/movies/books and allow them to find something that might suit their needs. While this might work for many users there will always be exceptions. For those few that enjoy rare books or indie games you can do more extensive searching. Due to the differing interests of the common consumer of media, we decided to include more than just one form of media. We decided on three of the major visual medias namely movies, books and computer games.

To make this possible we had to design multiple graphical interfaces as well as a database and of course all the code had to interact with each other.

The educational purpose of this project was to test and improve our group communication, programming, database design, as well as many other aspects of software engineering.

We were originally a group of 6 people with varying prior knowledge of the related subjects, given a few months to develop a decision support system

Background Information

Student group Mjolnir is a recently assembled group of six students working from Lindholmen campus.

Definitions, Acronyms and Abbreviations

The abbreviation DDS refers to Decision Support System.

The term “media” refers to both games/books/movies in this document.

The project

We would like to first introduce a little bit about the general background knowledge of our group, regarding the various aspects of programming. There are in total six group members in our group (Mjolnir), four of which did not have any previous knowledge about programming before the start of this academic school year. Only two of the group members had some previous experience, thus by definition there had to be some level of balance between the group's working strategy. This means that not only did we learn ourselves the fundamentals of a software development process, but also the basis of co-operation and interaction of group members that may as well face each other for the first time a few days before an assigned project.

The programming language in which we built this system is the Java language; it is specified in the project requirements set by our program. What's more, it was the language taught during the current semester, thus it seemed the default choice.

For most of us it was the first time to build a software system. For some, even the very first time to implement the simplest Java application with functionality. We did understand there would be plenty of difficulties throughout the whole process in the development of this project but we did not give up easily in any case. All of the group members have been making their best effort to give contributions in order to make everything work, for both the coding itself and the group dynamic.

System Definition

The purpose of our project was to build an application decision support system which is used to add reviews for different items belonging to books, movies and games; the inspiration was taken from the famous website Metacritic.

Our application was designed for system administrators and users to use. A log-in system was designed to protect the authority of the application and therefore an account must be created before a user can add any reviews or pictures on books, movies or games. The reviews will display promptly on the page after they have been sent by users, yet the pictures will be checked by the system administrators first. A rating system was designed for users to get a general understanding of the quality

and the content of certain items that they are searching for and also to evaluate it afterwards. Users will need to give scores that rank from 1 to 5 while they are adding reviews. The top list panel containing the four top rated items will be found on the main page of the system.

The problem Domain

Clusters



Figure 1: Clusters

The Figure 1 above shows the two main clusters that were considered for the design of the system, namely the administrators and the users.

Detailed Project Structure

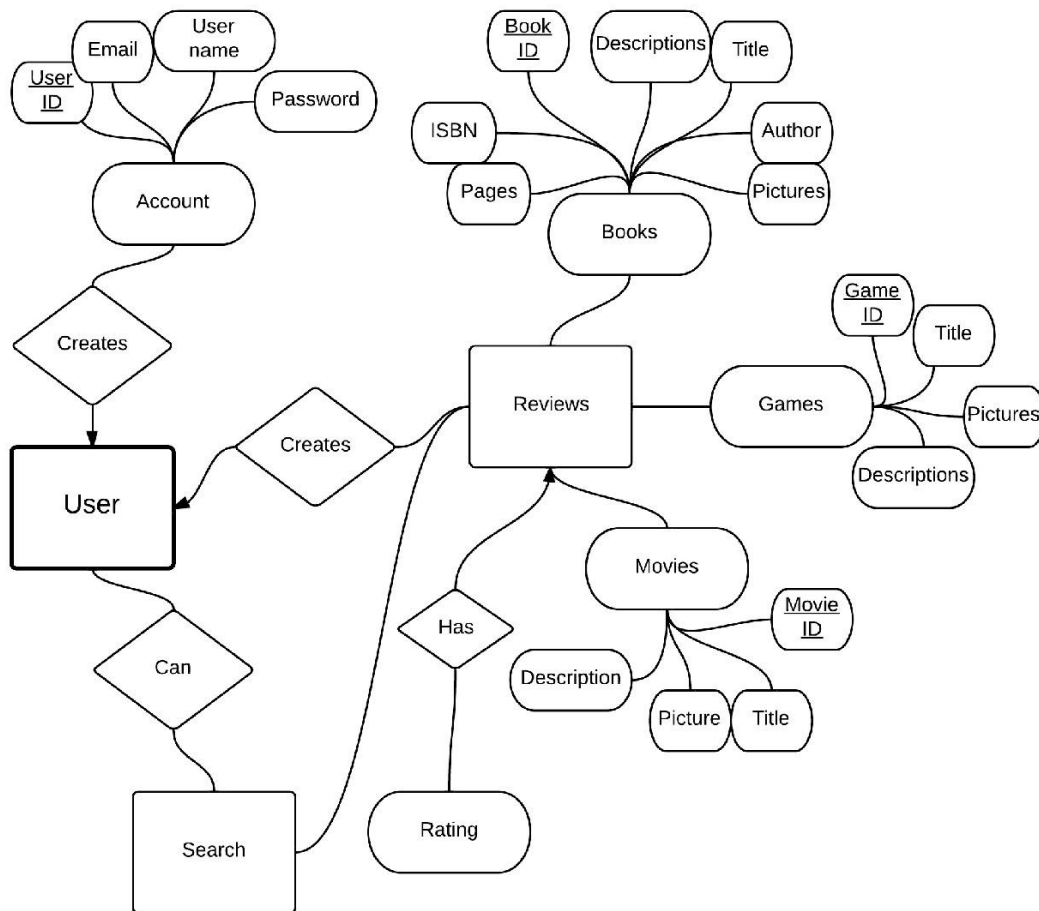


Figure 2: The user cluster

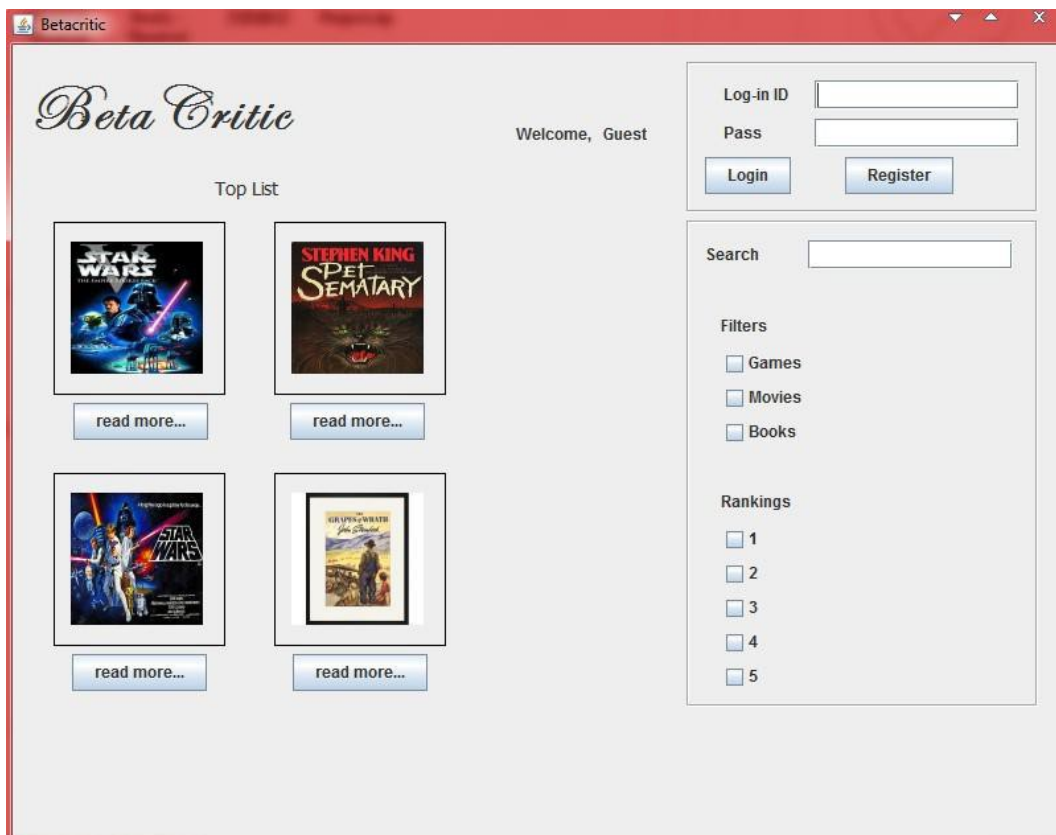
The user cluster is illustrated clearly in the figure 2 above; it covers almost all the functions that the system provides. Users are supposed to create user accounts, while all the required fields must be completed accordingly. Reviews can be searched or added on books, movies and games. Users can also choose to filter scores ranking from 1 to 5 and types of reviews while searching for the desired result.

Administrators do not have accounts, however they are able to access the database and have the right to delete or add users, reviews, items and approving pictures that are uploaded by the users.

Application Domain

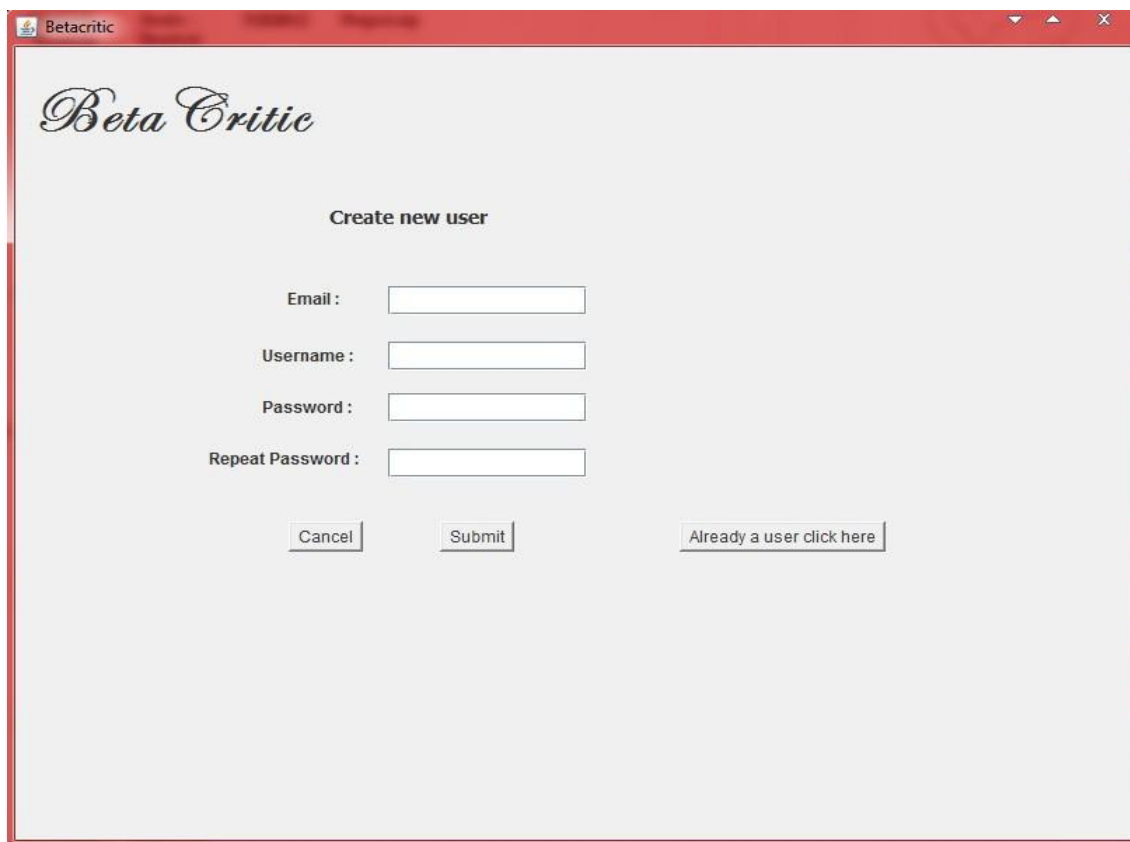
In this section, the detailed explanation of all functions offered by the system will be fully introduced. Furthermore, some specific examples of use cases will also be mentioned here.

Functionalities



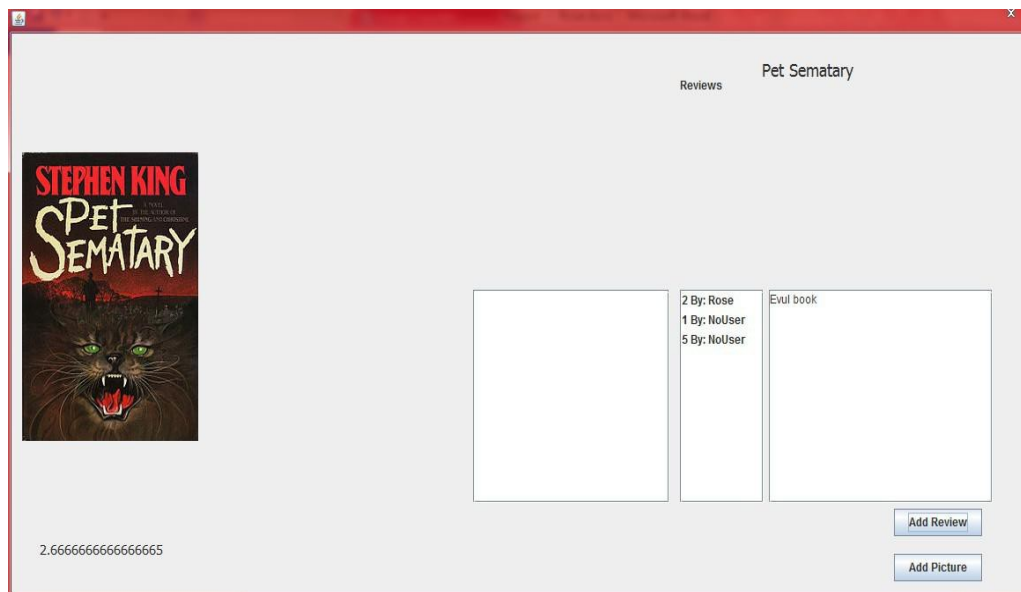
This is how the main page of the application looks like. On the left hand side you will find the top list; there is a fixed frame on the right hand side, which contains the log-in panel, the search field, the filter for different types of items and the rankings.

Clicking the register button on the log-in panel, the application will turn to the user creation page. This is where the user needs to go in order to create an account and log-in to the application.

The image shows a web browser window titled "Betacritic". The page has a light gray background and a red border. At the top left, the "Beta Critic" logo is displayed in a stylized, cursive font. Below the logo, the text "Create new user" is centered. There are four input fields arranged vertically, each with a label to its left: "Email :", "Username :", "Password :", and "Repeat Password :". Each label is followed by a white rectangular input box. At the bottom of the form, there are three buttons: "Cancel", "Submit", and "Already a user click here". The "Cancel" and "Submit" buttons are small and rectangular, while the "Already a user click here" button is slightly larger and has a thin border.

All four fields must be filled in before the submission, the username and the password must be at a length of more than five characters and the repeated password should match the password entered. Keep in mind that if you would submit without filling in all the information according to the requirements, the registration would not be successful and an error message will be shown.

After logging-in, clicking the read more button of any item on the top rated list, a pop-up page will be shown similar to the one below.

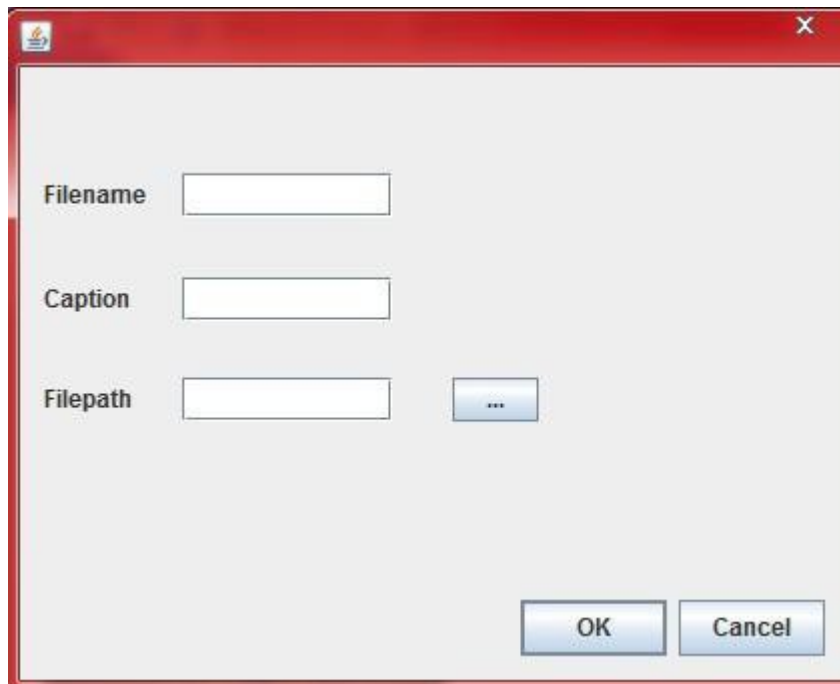


The left section of the interface shows the picture of the item and the right section shows the user reviews and the scores given.

Users can add reviews and give scores by clicking “Add Review”, or change pictures by clicking “Add Picture”. The picture that is uploaded will be checked by the system administrators and it will be decided whether it can replace the current one or not.



Users can choose to rank the item in reference from 1 to 5, leave a comment on the media and then send the review by clicking the button “Send Review”.



A Windows-style dialog box with a red title bar and a close button (X) in the top right corner. The dialog box contains three input fields: "Filename", "Caption", and "Filepath". The "Filepath" field has a small blue button with three dots (ellipsis) to its right. At the bottom right of the dialog box are two buttons: "OK" and "Cancel".

Here is the pop-up page for changing the picture of an item, clicking “OK” to send it. Users can either go back to the main page by clicking the “BetaCritic” sign or start searching for new media.

Structure and Organization

Usefulness and Feasibility

According to the stakeholders, the current Decision Support System provides useful solutions to the customer, while satisfying the functional and non-functional requirements that were set at the stage of planning the software. All five topics of feasibility were examined (Technical, Economic, Legal, Operational, Scheduling)¹ and the system proved to have high potential of success.

Specifically, the deadlines and the resources available for the project are rational, thus the implementation of the system within the current limitations is estimated as possible. What's more, the product will be prompt to changes, allowing the stakeholders to alter its features within limited time and cost.

Software Process Strategy

Our group followed a plan-driven development approach for some of the project's parts, while a more flexible strategy was used during the actual implementation of the project's coding. The initial thought of the project began with a feasibility study in order to assure that the resources available are enough to implement this project within the given constraints set by the stakeholders. *Requirements analysis and validation*² was the next step, so that, among other things, the project would not aim at unrealistic expectations. A thorough Project Plan was produced which included the project's requirements, a work breakdown structure and a risk management section. This was the base for our work and a way to plan ahead for our project.

The further implementation of the project followed an incremental approach, so as to be able to go back and forth in various stages of the process, in order to fix/alter functions and gain flexibility.

¹ http://en.wikipedia.org/wiki/TELOS_%28project_management%29

² Sommerville, Ian (2011). *Software Engineering* 9th edition. pg. 110

Team Organization

We, as a group began with assigned roles, although during the implementation process it was necessary to switch at times, partly due to our lack of experience with working in such a project, but also because it was the first time that we were working as a group together. The initial organizational structure as stated in our project plan was as follows:

Name	Title	Responsibility
Petroula Theodoridou	Project Manager	Managing the DSS project. Primarily responsible for organizing, motivating, reporting and tracking the project. Additionally, the project manager will also coordinate with the supervisor.
Natalie Davidsson	Front End Developer	The overall design of the interface of the application.
Calle Persson	Back End Developer	The coding and programming involved in the project.
Simon Lobo	Technician Manager	Monitoring the system for errors in performance. Responding to program error messages or bugs by finding and correcting them.
Peili Ge	Quality Manager	Responsible for creating quality control and assurance standards and keeping records throughout the project on these checks.
Yurou Zhao	Software architect	Making major design decisions. Decides technical standards, including software coding standards and platforms.

Since most of us did not have any prior programming knowledge nor did we knew each other beforehand, this project proved to be both challenging and enlightening in various aspects.

Detailed Individual Work

For the purpose of accurately credit the work done by each member throughout this project, it is essential to have this sub-section in our report. In order to better organise the current part, the individual work is going to be divided in categories; namely the Project Plan, the Coding and the Project Report.

Project Plan

1.1 Introduction - Petroula

21 Work Breakdown Structure - Petroula and Natalie

2.1.1 Team Division

2.1.2 Time Estimation and Schedule

22 Project Scope - Petroula

2.2.1 Functional Requirements - Calle, Simon, Peili, Rose

2.2.2 Non - Functional Requirements - Calle, Simon, Peili, Rose

23 Risk Management - Petroula

2.3.1 Risk identification and Analysis

2.3.2 Risk planning

2.3.3 Risk monitoring mechanisms

Document checking - Petroula and Natalie

Final document preparation – Petroula

Coding

Due to the fact that the coding part cannot be divided easily, the name of the member that mainly wrote each *class* is going to be displayed, but also with supplementary references to those people that added their custom code inside that *class*.

Default package

AddBook.java - **Simon**

AddMovieGame.java - **Simon**

AddPicture.java - **Simon**

AddReviewDialog.java - **Simon**

BetaCritic.java - **Petroula**

EditReview.java - **Simon**

LoginPanel.java - **Calle**

Supplementary code: jCheckboxes - **Peili**

mouse listeners for changing panels – **Petroula, Simon**

login with username and password - **Rose**

MainPage.java - **Petroula**

Supplementary code: show photos for the top 4 results – **Simon**

ReadMoreDialog.java - **Peili**

Supplementary code: getting reviews and ratings – **Calle**

show photo for each item - **Simon**

SearchPage.java - **Petroula**

Supplementary code: search function and clickable results - **Calle**

UserCreation.java - **Petroula**

Supplementary code: register new user - **Rose**

UserPage.java - **Petroula**

Supplementary code: display all the reviews from the current user - **Simon**

BetacriticEasyDatabase

BaseMedia.java - **Calle**

Book.java – **Calle**

DatabaseManager.java - **Calle**

DatabaseManagerMedia.java - **Petroura**

Supplementary code: add picture - **Simon**

Game.java - **Calle**

Movie.java - **Calle**

Review.java - **Calle**

Supplementary code: get username/title - **Simon**

User.java – **Calle**

Project Report

Project Overview - **Natalie**

Purpose of this document - **Natalie**

Acknowledgements - **Natalie**

Introduction - **Calle**

Background Information - **Calle**

Definitions, Acronyms and Abbreviations - **Calle**

The Project - **Rose**

System Definition – **Rose**

The Problem Domain - **Rose**

Clusters

Detailed Project Structure

Application Domain - **Rose**

Functionalities

Structure And Organisation - **Petroura**

Usefulness and Feasibility

Software Process Strategy

Team Organisation

Detailed Individual Work

Project Plan

Coding

Project Report

Schedule

Testing and Validation - Peili

Strengths vs Weaknesses - Peili

Evaluation - Peili

Changes to the Final Product - Natalie

Requirements - Natalie

Requirements Summary - Natalie

Major Challenges and Solutions - Simon

Decision Making

Choosing Design

Database

Editing the Interface

Combining Code

Prioritizing

Repeated Code

Conclusion and Future Work - Calle

Conclusion

Future Work

References - Petroula and Natalie

System User Manual - Simon

Combining sections - Natalie

Document checking - Petroula and Natalie

Final document preparation - Petroula

Schedule

We had group meetings scheduled usually twice a week, from the 21st of October, in order to work with the project. When there was much work to be done (important milestones ahead) we met more often and also spent more time on it. Aside from the group gatherings, all members of the group devoted individual time for this project, as it was planned.

Testing and Validation

We have successfully tested the usability of our BetaCritic project through Eclipse and Netbeans. The testing process is looking into the interaction between the initial design requirements and the object's participation.

Through the usability testing we are aiming at the following goals: a) a system that is easy to understand and handle by users of different level of expertise, b) a final product that is useful, thus a product that meets end-users' real needs and requirements and c) a bug/error-free system³.

In general, we tested the program part by part in order to find the errors and fix them. The different parts that we divided the application into, were the main page, the login panel, a search page, a user creation page, a user page, read more dialogs and add review dialogs. During the testing process, there are always errors engendered in the system somehow, maybe partly due to our unfamiliarity with the programming tools we were using and some code transferring problems. It turned out to be more challengeable than the initial design requirement goals we had set for the project. We had to fix errors again and again by overwriting the code and making changes to the design in order to find ways of implementing it better. Hence, the final product greatly meets the requirements we set and as far as the final version is concerned, it passed all tests without any issues.

Strengths vs Weaknesses

We have spent most of our time customizing the interface for our project; almost everyone got involved with it, mainly because most of the functions offered by the system are attached to the interface. Therefore, the variety of our interface design is our main strength. On top of that, one has to consider the usability of our project for the end users who need easy and quick search results of movies, books and games; we undoubtedly provide a well connected platform between potential users and sources. In addition, we have a good control of people management in our project. We have a set time table of having meetings and project work. We have always been able to keep

³ Steven G. *The Resonant Interface: HCI Foundations for Interaction Design*. Boston: Pearson/Addison Wesley, 2008. Print.

to each deadline and double tested our system functionality. We have also spent considerable time discussing the feedback given from our supervisor and consistently tried to improve ourselves. We used to distribute the work needed to be done by each group member and discussed about each other's individual progress, in order to give opinions and suggestions that would enable us to establish a better project. On the other hand, due to the lack of communication sometimes in the group, we have wasted time not focusing on discussing the project. This happened not only because that project was new for most of the people in the group, but also due to our lack of experience on the subject. We realize that these two factors may have affected the overall quality of the project. However, with the help from our supervisor and the more experienced group members, we balanced strengths and weaknesses to get the project done within the initial time constraints, met the requirements we set and managed to live up to our quality expectations.

Evaluation

As the main purpose of our project was to build a system which provides end-users with a connection to add reviews for different items (books, movies and games), as well as search information about it, we as a group believe that requirements are met and the final product is similar to our initial idea.

For the overall design, we have changed some parts of the structure for almost each page in the project. Compared to the initial design, we have changed the login panel to a set size and also the various interface pages to fit better the user needs and requirements.

We are to a high extent satisfied with the programming tools we chose and we may make small adjustments next time for better structure design and color combinations.

The whole program is well done; it fits greatly with the requirements plan and successfully delivers its main purpose and functions for end-users.

Changes to the final Product

There were some features of the developed software that were not included in the final product of the Java application. One of these features had to be the administration tools. The administration tools were intended to give trustworthy users administrator privileges in order to, for example protect against malicious users. On the other hand, it could be a money sink and a security risk. Many of the services that an administration tool offers, such as deleting and updating without restrictions, can easily be done in other ways (Database graphical interface) during the development phase. Another change to the final version was the decision not to include a user promotion system. Extra features were planned to be links to movie trailers and option to bookmark your favorite books, movies and games.

Requirements

Number	Requirements Description	Completed
1	Search Function	Yes
2	Ranking System	Yes
3	Administration tools	No
4	User creation System	Yes
5	User promotion System	No
6	Language: Java	Yes
7	Simplistic Design	Yes

Requirements Summary

Total number of requirements	7
Number of requirements implemented	5
Requirements partially fulfilled	0
Requirements not fulfilled	0
Requirements dropped	2

Major Challenges and Solutions

Starting with the fact that not all of us had much experience in programming and database management we knew that this would be a difficult project. We encountered some challenges that we at first found somewhat overwhelming. In this section we describe into detail the major challenges we faced in our project and how we overcame those in order to achieve our goals.

Decision Making

At the beginning of the project we encountered some issues that held us back. A couple of hours were lost the very first day when we were to decide on what to make the DSS about. When we managed to decide on a DSS we had trouble deciding what sort of reviews the program should handle. We would have lost more time if it were not for our supervisor, who pointed out that our idea was good and that it did not matter what sort of media we would choose to review.

Choosing Design

Every day was a brand new challenge; sometimes we would lose an hour or two just by discussing the day's work and then deciding how to start. We designed several model interfaces and as soon as we decided which one to use we stumbled upon an issue. What IDE should we use? Since we all had just started learning Java, we wanted to use the recommended IDE: Eclipse. Creating an interface with Eclipse would take a lot of time and using a window-builder for this IDE, proved to be a rather difficult task. One of the members knew that Netbeans IDE comes with an integrated window-builder, so some people started using it while others were still trying to make Eclipse work, but to no avail. With Eclipse's window-builder not working we decided to simply use Netbeans, although we wanted to code using Eclipse because we were accustomed to using it. We also found Netbeans less appealing and difficult to understand; of course, we could have taken the time to learn it, but at that time we felt that we had lost enough time so we continued with the original plan: designing in Netbeans and coding in Eclipse.

Database

After creating the database using MySQL some of our team-members had trouble installing and using this relational database management system (RDBMS). So at the beginning only four of our six members had complete access to the database. With the help of other classmates and supervisors the two other members managed to fix their issues. Later, one of our members bought a new computer and we could never make the database run on the new computer. This meant that the person had to use someone else's computer to do his/her work. So the work of these two people would take twice as long to complete.

Editing the interface

After we had been working with the design for some time we were ready to start coding, but we found an issue. At times, while coding, we would want to change the design, just a little. This required changing the design in Netbeans and then copying and pasting the necessary code into Eclipse. So we had to save our code on a notepad, copy the design from Netbeans, paste it into Eclipse and finally carefully re-enter our code. This was not much of a problem but it was tedious. As stated before, it could have been avoided by learning to work entirely in either Netbeans or Eclipse.

Combining code

Since we did not follow our supervisor's advice to use Git in order to work on the same project from different computers, we had trouble combining our code. We mainly used Facebook for uploading the source code in the group, in order to be accessible for other members. Consequently, when members would make a change to a Java-class they would upload it and the rest would have to download it. If more people had been working on the same class and one had not yet uploaded the personal work, others would have to wait and combine the uploaded class from the other member. This would often be annoying work. Moreover, if we made changes and uploaded them from home it would make it hard to help the other members if they encountered problems.

Prioritizing

During the first couple of weeks most of us would work on tasks that were not essential for a beta product. This was mostly due to the fact that we could not prioritize our tasks. Some of us would also work on the same tasks together in order to help each other, which was a good idea but we were still unorganized. It was not until Thor, our supervisor, showed us a way to organize and prioritize tasks in order to work effectively. He specifically helped us in the way of gathering us together and making us prepare a list of the things that needed to be done. After the list was completed he told us each task should be assigned a number from one to ten, one meaning not essential and ten meaning that “it should have been done yesterday”. Seven was the barrier between the beta and the final version; a number above seven had to be done for the beta-version. Each task was voted for by all members with majority rule.

Repeated code

When we started coding we did not think much of it, but as we progressed through the Java lectures we learned that the code should be easy on the eyes. We noticed that we had failed to follow that rule, having much repeated code in our application. This was probably due to our desire to complete a task quickly and move on to the next one. Gradually, we managed to fix all these mistakes. With the help of the lectures we were able to understand most, although it needed great effort and patience from our side. We might have missed some code and some might have been too difficult considering our level of expertise, however we feel like winners at the end of this process.

Conclusion and Future Work

Conclusion

In conclusion working on Betacritic has been a very good learning experience. Not only have we improved our programming skills but we have also improved our ability to work in a group.

During the project we had extensive use of form designers (Netbeans) and database managers (phpMyAdmin). PhpMyAdmin worked very well without any major problems. Netbeans on the other hand resulted in plenty of unnecessary work due to us editing the code in Eclipse. The result was that every time the interface needed to have any major changes we had to revert to an old version compatible with Netbeans and redo the code. This issue could have been solved by using Netbeans solely or by using an Eclipse form designer. On the other hand Netbeans allowed us to quickly produce interfaces that would otherwise take a significant amount of time to code. We also faced some non-technical problems. One of these was our lack of effective preparation before the initial presentations. We did however improve on this during the project.

Furthermore, there were some things that went really well with the project and one of them was our ability to communicate. We did not have many misunderstandings and when there was one it was always possible to quickly find a solution. Everyone in the group was very open to expressing their thoughts and ideas.

Even if there were some challenges and issues with the project and working as a group, we learned from these mistakes and we will make extensive effort to prevent similar issues in future projects.

Future Work

For future projects we would attempt to get started with the work load earlier, be more organized and set up a team structure earlier on in the project.

We could also set up rules in order to save time: for example we can set up a rule that if something takes longer than one hour to decide, we will just pick the first alternative that comes to mind in order to save time.

Dividing the work tasks will be easier due to the fact that everyone's programming skills have developed significantly making more of the tasks doable by more members of the group.

We should also read up on how the different tools are used together before we start working on any project, since our experience showed that Eclipse and Netbeans are quite tricky to use together.

References

Sommerville, Ian. Software Engineering. 9th ed. Wokingham, England: Addison-Wesley Pub., 1996. Print.

Wikipedia contributors. "TELOS (project management)." Wikipedia, The Free Encyclopedia, 14 Aug. 2012. Web. 15 Jan. 2014.

Steven G. The Resonant Interface: HCI Foundations for Interaction Design. Boston: Pearson/Addison Wesley, 2008. Print.

System User Manual

Setup:

Step 1: Upload the BetaCritic.sql to a MySQL server.

Step 2: Change the settings.txt file to setup your connection. Default setting is root with no password.

Usage:

Start the executable JAR file and you're ready to go.