Football Manager

Analysis and Design Document

Student: Milas Bogdan-Adrian

**Group:30233**

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| <06/04/2017> | <1.0> |  | Milas Bogdan-Adrian |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

I. Project Specification 4

II. Elaboration – Iteration 1.1 4

1. Domain Model 4

2. Architectural Design 4

2.1 Conceptual Architecture 4

2.2 Package Design 4

2.3 Component and Deployment Diagrams 4

III. Elaboration – Iteration 1.2 4

1. Design Model 4

1.1 Dynamic Behavior 4

1.2 Class Design 4

2. Data Model 4

3. Unit Testing 4

IV. Elaboration – Iteration 2 4

1. Architectural Design Refinement 4

2. Design Model Refinement 4

V. Construction and Transition 5

1. System Testing 5

2. Future improvements 5

VI. Bibliography 5

# Project Specification

Scopul acestui “Football Manager” este de a produce soft care manageriaza si gestioneaza meciurile unui turneu de fotbal care contine mai multe echipe.Mai concret ,modul in care echipele se vor duela intre ele , locul de desfasurare al meciurilor. dar si altele. Datele despre turneu si echipele participante vor fi stocate intr-o baza de date. Detaliile cu privire la modul în care Football Managerul își îndeplinește actiunile sunt detaliate în cazurile de utilizare și specificatiile suplimentare.

Aplicatia va trebui sa aiba 2 tipuri de utilizatori (un administrator si un utilizator obisnuit) care vor trebui sa se autentifice printr-un nume si printr-o parola.Utilizatorul normal se va putea loga , va putea cumpara bilete , chema alti prieteni sau comenta meciurilela care participa.Administratotul , in schimb , se va putea si el loga ,va putea adauga un turneu ,va putea tipari bilete si ocazional poate active anumite oferte pentru primii client.

# Elaboration – Iteration 1.1

# Domain Model

Modelul nostru va fi structurat(aproximativ) cum arata urmatoarea diagrama de clase:



# Architectural Design

## Conceptual Architecture

Aplicatia noastra va folosi MVC si Layers pattern.

MVC pattern:

-Model va fi folosit pentru structura aplicatiei.De exemplu : felul in care se reprezinta clasele (cu metode , constructori) sau un table din baza de date.

-View si Controller vor fi utilizate pentru tot ce vede cel ce foloseste aplicatia.

Layers pattern:

-Business Layer : aici vor fi incluse functionalitatile de care va avea nevoie aplicatia.

-DataAcces Layer : va accesa inregistrarile din baza de date pentru a se putea efectua operatiile generale .

-Database Layer : va face legatura cu baza de date.

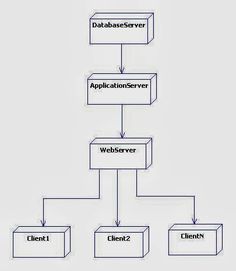
-Presentation Layer : va reprezentata interfata pe care utilizatorul o va putea vedea.

## Package Design

*[Create a package diagram]*

## Component and Deployment Diagrams

Deployment Diagram



Component Diagram



# Elaboration – Iteration 1.2

# Design Model

## Dynamic Behavior

*[Create the interaction diagrams (1 sequence, 1 communication diagrams) for 2 relevant scenarios]*

## Class Design

*[Create the UML class diagram; apply GoF patterns and motivate your choice]*

# Data Model

*[Create the data model for the system.]*

# Unit Testing

*[Present the used testing methods and the associated test case scenarios.]*

# Elaboration – Iteration 2

# Architectural Design Refinement

*[Refine the architectural design: conceptual architecture, package design (consider package design principles), component and deployment diagrams. Motivate the changes that have been made.]*

# Design Model Refinement

## *[Refine the UML class diagram by applying class design principles and GRASP; motivate your choices. Deliver the updated class diagrams.]*

# Construction and Transition

# System Testing

*[Describe how you applied integration testing and present the associated test case scenarios.]*

# Future improvements

*[Present future improvements for the system]*

# Bibliography