Vertraulich

IPA2018

autor(en) : Cassandra Corrodi

dokumentennummer : CCT-FFF-XXXX

version : 1.0

status : Draft

quelle : Atos

dokumentendatum : 24 Oktober 2018­

­

­

anzahl der seiten : 1

owner : Cassandra Corrodi

© Copyright 2018, Atos AG Alle Rechte vorbehalten. Reproduktion von Teilen oder dem Gesamten ist ohne vorherige schriftliche Genehmigung des Urhebers untersagt. Für Fragen oder Anmerkungen zu diesem Dokument wenden Sie sich bitte an Atos, 0795893817.

# Inhaltsverzeichnis

[1 Teil 1: Umfeld und Ablauf 4](#_Toc528163134)

[1.1 Einleitung 4](#_Toc528163135)

[1.2 Beteiligte Personen 5](#_Toc528163136)

[1.2.1 Durchführung der IPA 5](#_Toc528163137)

[1.2.2 Kandidat 5](#_Toc528163138)

[1.2.3 Fachvorgesetzter 5](#_Toc528163139)

[1.2.4 Ausbildungsverantwortlicher 5](#_Toc528163140)

[1.2.5 Experte 5](#_Toc528163141)

[1.2.6 Zweit Experte 5](#_Toc528163142)

[1.3 Aufgabenstellung 6](#_Toc528163143)

[1.4 Projektorganisation 6](#_Toc528163144)

[1.5 Vorkenntnisse 6](#_Toc528163145)

[1.6 Vorarbeiten 6](#_Toc528163146)

[1.7 Firmenstandards 6](#_Toc528163147)

[1.8 Zeitplan 7](#_Toc528163148)

[1.9 Arbeitsprotokoll 8](#_Toc528163149)

[2 Teil 2: Projekt-Dokumentation 18](#_Toc528163150)

[2.1 Informieren 18](#_Toc528163151)

[2.2 Planen 18](#_Toc528163152)

[2.3 Entscheiden 18](#_Toc528163153)

[2.4 Realisieren 18](#_Toc528163154)

[2.5 Kontrollieren 18](#_Toc528163155)

[2.6 Auswerten 18](#_Toc528163156)

[3 Teil 3: Anhang 19](#_Toc528163157)

[3.1 Quellenverzeichnis 19](#_Toc528163158)

[3.2 Abbildungsverzeichnis 19](#_Toc528163159)

[3.3 Tabellenverzeichnis 19](#_Toc528163160)

[3.4 Glossar 19](#_Toc528163161)

[3.5 Programmcode 19](#_Toc528163162)

Änderungshistorie

| Version | Datum | Beschreibung | Autor(en) |
| --- | --- | --- | --- |
| 0.1 | 21.11.2018 | Dokumentstruktur erstellt, Einleitung verfasst, mit Planung begonnen, Zeitplan vorbereitet | Cassandra Corrodi |
| 0.2 |  |  | Cassandra Corrodi |
| 0.3 |  |  | Cassandra Corrodi |

# Teil 1: Umfeld und Ablauf

## Einleitung

Im Rahmen der Probe IPA, welche bei Atos stattfindet wird das Spiel Yatzy realisiert. Dieses Projekt ist eine Aufgabenstellung von der vergangenen Swiss Skills Meisterschaft. Die Aufgabe ist es nun, diesen Auftrag mit den technologien zu realisieren, mit denen auch die eche IPA geschrieben wird. Zudem muss die Applikation sauber getestet werden und es muss eine gute Dokumentation vorgewiesen werden.

## Kandidatin

Im Sommer 2015 habe ich meine Lehre zur Informatikerin Fachrichtung Applikationsentwicklung bei Siemens Schweiz gestartet. Im ersten Lehrjahr habe ich dort die Basisausbildung absolviert. Alle Informatiklehrlinge vom gleichen Lehrgang waren während dieses Jahres zusammen, und mussten theoretische wie auch praktische Arbeiten lösen.

Im zweiten Lehrjahr wurden alle in die Abteilungen verteilt. Ab dann habe ich bei Atos gearbeitet und konnte in den Abteilungen praktische Erfahrungen im Bereich C# und Angular sammeln. Zudem durfte ich auch eine Woche im First-Level-Support arbeiten, bei dem ich auch nützliche Erfahrungen mit Kunden sammeln und verschiedene Probleme lösen konnte.

Im Moment arbeite ich im SWP Team, und implementiere mit ASP.NET MVC.

## Beteiligte Personen

### Durchführung der Probe IPA

Lehrbetrieb: Atos AG  
Adresse: Freilagerstrasse 28  
PLZ/Ort: 8047 Zürich

### Kandidatin

Name: Cassandra Corrodi  
Adresse: Alte Bergstrasse 7  
PLZ/Ort: 8707 Uetikon am See  
Telefon: 079 517 38 89

### Fachvorgesetzter

Name: Patrick Maurer  
Adresse: Freilagerstrasse 28  
PLZ/Ort: 8047 Zürich  
E-Mail: patrick.maurer@atos.net

### Ausbildungsverantwortlicher

Name: Jonas Knoll  
Adresse: Freilagerstrasse 40  
PLZ/Ort: 8047 Zürich  
Telefon: 058 558 38 58  
E-Mail: jonas.knoll@siemens.com

### Experte

Name: Torben Dziuk  
Adresse: Freilagerstrasse 28  
PLZ/Ort: 8047 Zürich  
E-Mail: torben.dziuk@atos.net

## Aufgabenstellung

Das Ziel dieses Projektes ist es, eine Probe IPA zu schreiben, damit der Kandidat für die echte IPA welche im Frühjahr stattfinden wird vorbereitet ist. Die Aufgabe wird sein, eine Aufgabe von den Swiss Skills in vier Tagen mit dazugehöriger Dokumentation zu realisieren.

Bei den Swiss Skills wird hauptsächlich die Oberfläche der Applikation bewertet. Das heisst wenn man die Applikation bedient, sollte diese einfach funktionieren. Dabei spielt es keine Rolle ob Clean Code Prinzipien oder eine saubere Implementierung generell eingehalten wurde. Zudem muss die Applikation nicht einmal getestet werden. Bei den Swiss Skills muss auch keine Dokumentation geschrieben werden.

Bei der Probe IPA wird es anders sein, denn es kommt nicht nur auf die Oberfläche an. Die Applikation muss zuerst sauber geplant werden, bevor man mit der Implementation beginnt. Während der Implementation ist es wichtig, den Code so sauber wie möglich zu schreiben, denn dieser kommt in den Anhang am Schluss der Doku.

Zudem muss die Applikation auf Herz und Nieren geprüft werden, damit die Qualität gewährleistet werden kann. Um das zu erreichen, werden die Testfälle bereits am Anfang des Projektes definiert.

Am Ende dieser Probe IPA, sollte die Kandidatin besser vorbereitet sein für den Lehrabschluss, und sich schon an die Routine von so einem Projekt gewöhnen.

**Ausgangslage**

Yahtzy ist ein alter Spielklassiker, an welches sich vielleicht noch einige Leute erinnern können.

Früher hat man es meist mit einem Block gespielt, bei dem man die Ergebnisse direkt eingetragen hat. Da wir uns nun im Jahr 2018 befinden, und nicht mehr alle Lust haben, die Ergebnisse von Kopf auszurechnen und auf einem Papierblock zu schreiben, wird das Spiel nun als Webapplikation realisiert.

Mit diesem Programm können zwei Spieler zusammen an einem Rechner sitzen und spielen. Das Programm übernimmt die Rechenaufgaben und das Würfeln, während die Spieler spass beim spielen haben sollen.

### Spielregeln Yatzy

Da auf dem Projektbeschrieb von den Swiss Skills nicht so gut beschrieben ist, wie das Spiel funktioniert, wurde im Internet recherchiert, wie das Spiel genau funktioniert. Das Spiel ist auch unter dem Namen Kniffel bekannt.

**Spielregeln**Bei Yatzy geht es darum, mit den 5 Würfeln die Kombinationen zu erwürfeln, die in der Spieltabelle vorgegeben sind. Gespielt werden 13 Runden, da es 13 verschiedene Kombinationen gibt. Jeder Spieler darf pro Runde dreimal hintereinander würfeln.

Nach jedem Wurf muss er entscheiden, welche Würfel er "stehen" lässt. Ein Würfel wird durch Klick auf das rechts neben ihm befindliche Kästchen fixiert. Es erscheint ein Häkchen, außerdem wird der Würfel abgeblendet. Einmal stehengelassene Würfel können später durch einen erneuten Klick wieder "befreit" werden.

Ist bereits der erste Wurf perfekt, muss natürlich nicht erneut gewürfelt werden. Spätestens nach dem 3. Wurf muss jedoch die entstandene Würfelkombination in eines der 13 Felder eingetragen werden. Danach ist der nächste Spieler an der Reihe.

Besonders zu Beginn gibt es zumeist mehrere Möglichkeiten, eine Würfelkombination in die Tabelle einzutragen. Ist jedoch ein Feld bereits belegt, so kann der betreffende Eintrag nicht mehr korrigiert werden. Lässt sich eine gewürfelte Kombination in keines der ersten 12 Felder eintragen, so hat der Spieler einmalig durch das Feld "Chance" die Möglichkeit, einfach die Summe der 5 Würfel einzutragen. Ist auch dieses Feld schon belegt, muss ein freies Feld "gestrichen" werden. In diesem Fall werden 0 Punkte vergeben.

Sind alle Felder belegt, ist das Spiel zu Ende. Gewonnen hat der Spieler mit den meisten Punkten.

(Quelle: http://www.2meter3.de/yatzy/rules.html)

### Projektantrag

Der Projektantrag ist schon genau vorgegeben. Folgende Punkte sind also während der Implementation dieses Projektes zu beachten.

**Projekt**

Die Aufgabe ist es Yatzy für zwei Spieler zu implementieren. Zudem muss eine Datenbank erstellt werden, bei der die Highscores gespeichert werden können.

**Gameplay**

Die Spieler würfeln abwechselnd mit fünf Würfeln. Nach jedem Wurf wählt der Spieler, welche Würfel er behalten und welche er erneut würfeln möchte. Ein Spieler kann einen Teil oder alle Würfel in einem Zug bis zu zweimal wiederholen. Der Spieler muss in jeder Runde eine Punktzahl in eine Score-Box eingeben. Das Spiel endet, wenn alle Punkteboxen verwendet werden. Der Spieler mit der höchsten Gesamtpunktzahl gewinnt das Spiel.

**Scoring**

Das Scoring ist schon genau auf der Aufgabenstellung von den Swiss Skills definiert. Man sieht auf den folgenden beiden Screenshots das Scoring. Das Scoring in diesem Projekt kann vom Spieler ziemlich selbstbestimmt gewählt werden, denn die Spieler entscheiden selber wie das Scoring ausfallen wird.

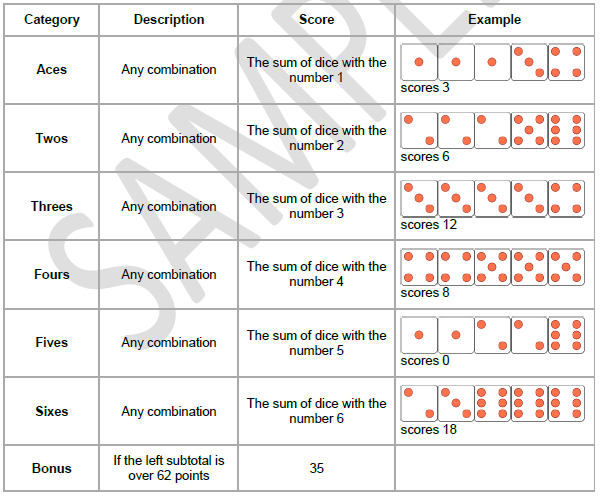


Abbildung 1: Scoring Zahlen einzeln

Auf der Abbildung sieht man das Scoring für die verschiedenen Würfelzahlen. Wie man sieht, werden diese, unabhängig davon, wie die Kombination ist, zusammengezählt. Jeder Spieler sollte wählen können, welche Würfel er behalten möchte, und welche er nochmals würfeln möchte.Die Spieler dürfen pro zug höchstens zwei Mal würfeln.

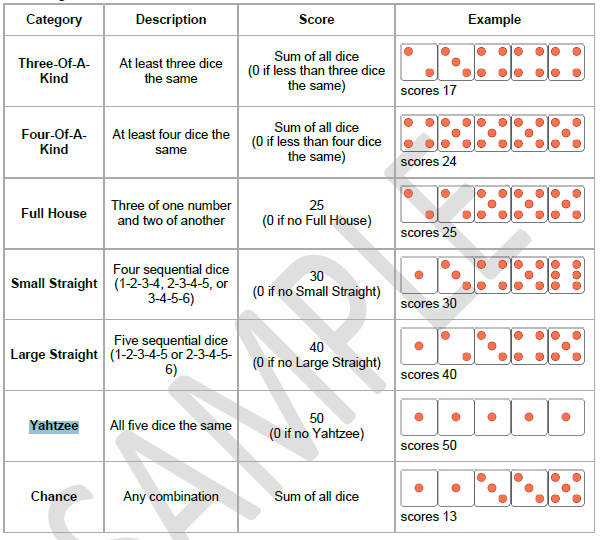
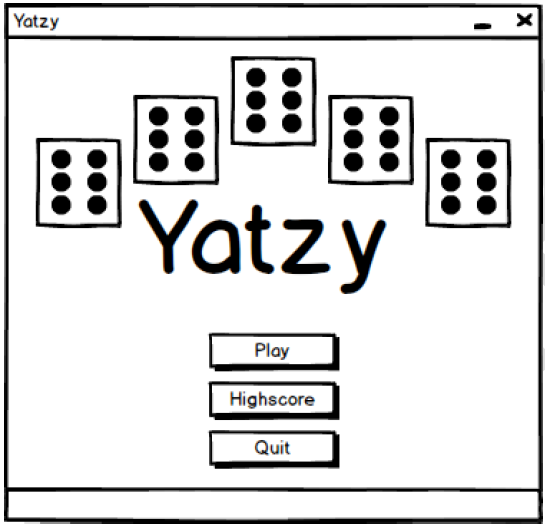


Abbildung 2: Scoring Zahlen Spezialfälle

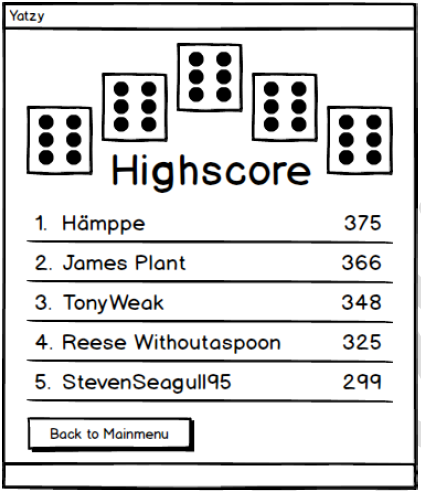
Auf dieser Abbildung sieht man, wie das Scoring ist, wenn man einen Spzialfall gewürfelt hat. Auf der rechten Spalte ist jeweils ein Beispiel ersichtlich, wie das Scoring je nach Fall ausfällt.

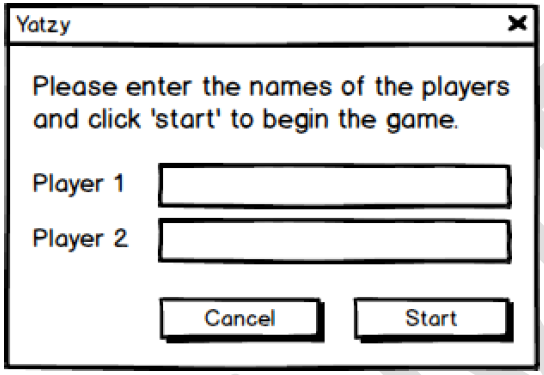
### Mockups

Das Frontend dieses Projektes ist bereits mehr oder weniger vorgegeben. Im Projektantrag der Swiss Skills befinden sich schon Mockups. Da ein Webprojekt erstellt wird, werden alle Fenster die gleiche grösse haben, und nicht so wie auf den Mockups alle anders sein.

**Startpage**

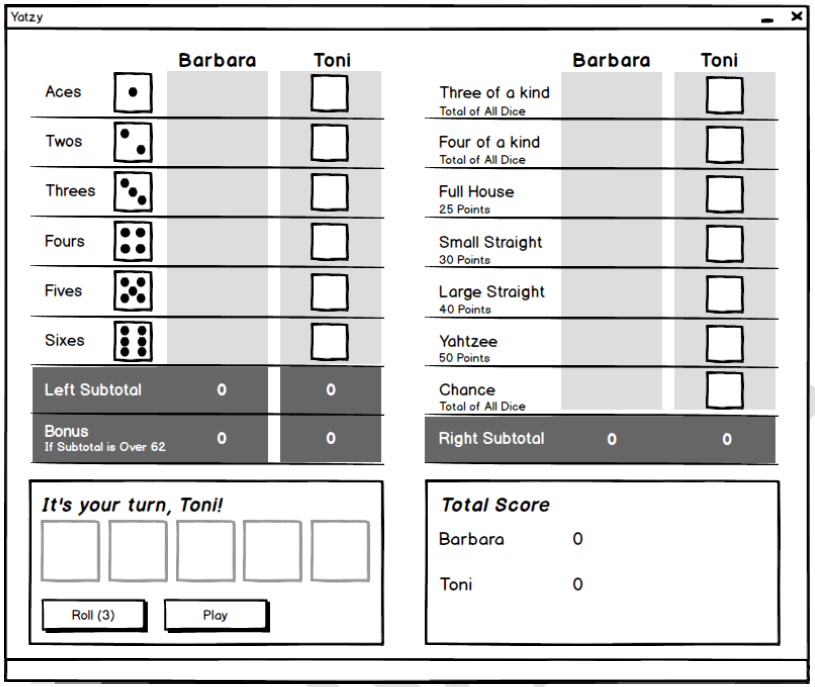
Auf der Startpage kann man ein Spiel starten, die Highscoreliste anschauen oder die Applikation beenden. Die Startpage sollte nur als Navigation dienen.

**Highscore List**auf der Highscore List sind die fünf besten Highscores aufgelistet. Von hieraus sollte es einen Button geben, mit dem man zurück auf das Hauptmenü, beziehungsweise Startpage geleitet wird. Diese Liste sollte automatisch aktualisiert werden, wenn jemand einen neuen Highscore erzielt, oder unter den neuen Top fünf gehört.

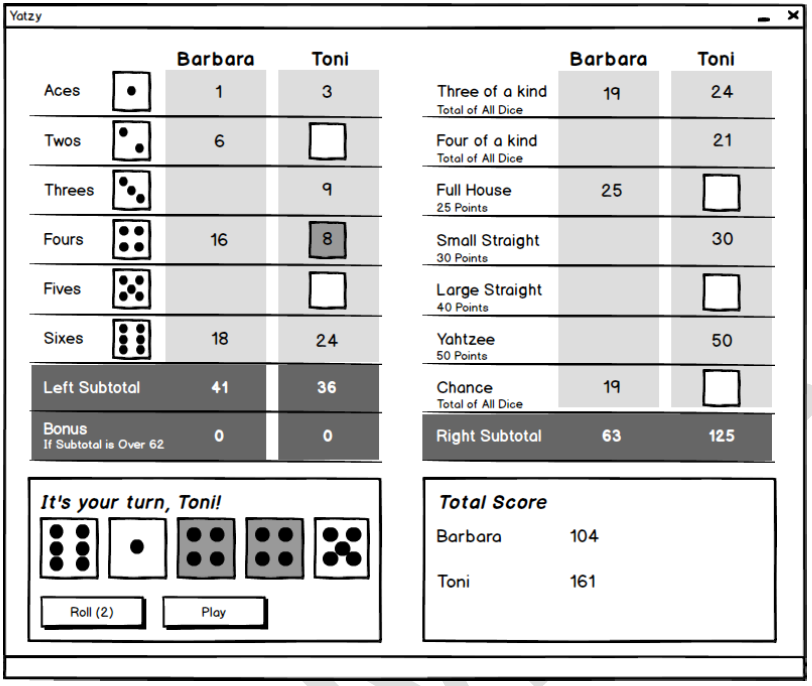
**Players**Wenn auf der Startpage auf «Play» geklickt wird, kommt man in eine View, bei der zwei Player erstellt werden müssen. Hier sollte eine Validierung der Inputs stattfinden, denn sie dürfen weder leer sein, noch mit Whitespaces gefüllt sein. Wenn eine fehlerhalte Eingabe gemacht wurde, darf kein neues Spiel gestartet werden und eine Error Message sollte aufploppen.  
Sonst können die Spieler einen beliebigen Namen wählen solange diese eine nicht zu lange Anzahl von Zeichen aufweist.

Wenn man auf den Start Button klickt, beginnt das Spiel. Bei Klick auf Cancel wird der Benutzer zurück auf das Startmenü geleitet.

**Game**

Hier sieht man das eigentliche Interface vom Spiel. Es hat zwei Spalten, die eine für das normale Scoring und die Andere für das spezielle Scoring.

Man sieht zwei Spalten. Für jeden Spieler eine. Der Spieler welcher gerade am Zug ist, sollte seine Spalte bearbeiten können. Unter der Spalte hat es einen Zähler welcher das Subtotal von der Spalte ausrechnet. Wenn ein Spieler einen Bonus gewonnen hat, wird dieser in einem Zähler unter dem andere Zähler zusammengerechnet.



Unter den beiden Spalten sieht man zwei Boxen. In der einen sieht man die Würfel und kann seine Würfe per Button kontrollieren. In der anderen sieht man den Total Score von beiden Spielern.

Wenn das Spiel beendet ist, wird man zurück auf die Startpage geleitet.

### Anforderungen des Arbeitgebers

### Technologiestack

IDE: Visual Studio 2017

Datenbank: SQL Server Management Studio

Frontend: Razor

Backend: ASP:NET MVC

Versionierung: GitHub undTortoiseGit

## Projektorganisation

### Projektmanagement Methode

Viele Projektmanagement Methoden sind eher darauf ausgerichtet, dass man diese im Team verwenden kann. Deswegen habe ich mich entschieden die Probe IPA mit IPERKA zu realisieren, da es sich sehr gut dafür eignet, wenn man alleine arbeitet.  
Alle Arbeitsschritte können gut in die verschiedenen Bereiche des IPERKAs geteilt werden, denn die verschiedenen Phasen sind so genau beschrieben, dass fast nichts schief gehen kann.

Ziel der Planung ist es, einen Lösungsweg für das Endprodukt zu erstellen. Hier müssen die Mittel und Arbeitsschritte genau geplant werden. Zudem muss hier definiert werden, wann was gemacht werden soll.

Zum Schluss wertet man aus, wie die Arbeit gegangen ist. Welche Schritte sind mir gelungen? Welche eher nicht? Wichtig dabei ist es, Erkenntnisse für das nächste Projekt zu sammeln.

Das ist der zeitaufwändigste Schritt von allen, denn die geplanten Arbeitsschritte müssen ausgeführt werden. Zudem muss man auch das gemachte dokumentieren.

Wenn die Arbeit erledigt hat, muss man diese kontrollieren. Man muss schauen ob alles erledigt wurde, und die Qualität des Programmes gewährleistet ist.

Nach der Planung muss definitiv entschieden werden, wie die Arbeitsschritte ausfallen werden. Dazu gehören auch Technologien und Wege, welche nicht schon im Voraus definiert wurden.

Man muss den Auftrag verstehen und sich ein Bild des zu ereichenden Ziels machen. Es müssen Abklärungen gemacht werden bis der Auftrag verstanden worden ist.

Abbildung 3: IPERKA

## Vorkenntnisse

Folgende Vorkenntnisse wurden in der Lehre gesammelt:

* ASP.NET MVC (6 Monate)
* SQL (Erfahrungen aus dem Basislehrjahr)
* Visual Studio mit ReSharper (3 Jahre)
* Git & TortoiseGit (2,5 Jahre)
* Angular Typescript (1 Jahr)
* Visual Studio Code (1 Jahr)
* C# (1 Jahr)
* jQuery (3 Monate)

## Vorarbeiten

Um während des Projektes nicht in Stress zu geraten, wurden ein paar Vorarbeiten geleistet. Diese sind auf der folgenden Liste aufgelistet.

* Die grobe Gliederung dieser Dokumentation wurde schon erstellt. Jede IPA Dokumentation ist anders, doch es gibt Bulletpoints welche alle Dokumentationen brauchen.
* Die IPERKA Abbildung wurde schon erstellt, damit während der Arbeit nicht so viele Grafiken erstellt werden müssen
* Text über Kandidatin wurde geschrieben.
* Liste der Beteiligten Personen wurde erstellt.

## Firmenstandards

Bei diesem Projekt stellt die Firma keine besonderen Standards. Tools und Technologie konnte selbst bestimmt werden.  
Atos hat ein Standard Template für Dokumentationen. Dieses wurde verwendet um die Dokumentation dieses Projektes zu schreiben.

## Zeitplan

### Meilensteine

Für das Projekt wurden Meilensteine festgelegt, welche die signifikanten Fortschritte eines Projektes darstellen. Da in diesem Projekt mit IPERKA gearbeitet wird, werden die Meilensteine nach jedem IPERKA Schritt gesetzt.

|  |  |  |
| --- | --- | --- |
| Schritt | Meilenstein | Datum/Zeit |
| I | Ich beginne die Arbeit und habe mich über verschiedene wesentliche Punkte informiert. Nach dem Informieren muss mir klar sein, was alles vorhanden ist, und was ich zu berücksichtigen habe während des ganzen Projektes. | 21.11.2018 um 16:30 Uhr |
| P | Ziel der Planung ist es, dass schon im Voraus überlegt wird, wie das Programm realisiert werden sollte. Die Planung ist ein Schritt, welcher wichtiger ist als die meisten denken. Denn hier ist das entscheidende, dass man während der Realisation nicht mehr so viel überlegen muss.  In diesem Schritt möchte ich Die Softwarearchitektur, die Views und die Datenbank geplant haben. | 22.11.2018 um 17:00 Uhr |
| E | Vor der Realisation sollte entschieden werden, wie die nicht vorgegebenen Schritte des Projektes vorgenommen werden sollten. Hier sollte dokumentiert sein, wie die Entscheidung gefallen ist. | 23.11.2018 um 12:00 Uhr |
| R | Bei der Realisation sollte das gesamte Projekt realisiert werden. Der Meilenstein ist zu Ende, wenn das Projekt ganz fertiggestellt ist. | 28.11.2018 um 12:00 |
| K | Nachdem das Projekt realisiert ist, muss es kontrolliert werden, weil die Qualität des Programmes gewährleistet werden muss. Ziel dieses Meilensteines ist es, Die geplanten Tests durchgeführt zu haben und die Ergebnissen sollten notiert werden. | 28.11.2018 um 17:00 |
| A | Ich schreibe die Reflexion und erstelle den Anhang. Somit wird die Dokumentation abgeschlossen, was auch der letzte Meilenstein dieses Projektes ist. | 28.11.2018 um 12:00 Uhr |

Tabelle 1: Meilensteine



## Arbeitsprotokoll

|  |  |  |  |
| --- | --- | --- | --- |
|  | Tag 1 (1/2 Tag) 21.11.2018 | |  |
| Soll | | Ist | |
| Projekt beginnen | | Erledigt | |
| Yahtzy studieren | | Erledigt | |
| Dokumentstruktur erstellen | | Erledigt | |
| Einleitung verfassen | | Erledigt | |
| Projektantrag zusammenfassen | | Aufgrund des Zeitmangels habe ich den Projektantrag nicht mehr zusammenfassen können. | |
| Zeitplan vorbereiten | | Erledigt | |
| Positives (Was lief gut?) | | | |
| Heute habe ich damit begonnen das Projekt zu realisieren. Dabei habe ich die Dokumentstruktur mit allen nötigen Kapiteln erstellt. Zudem habe ich mich über das Projekt informiert. Dabei habe ich die Aufgabenstellung und das Spiel Yatzy genau studiert. | | | |
| Negatives (Wo gab es Probleme?) | | | |
| Ich konnte den Projektantrag nicht verfassen, denn es stand mir nur ein halber Tag zur verfügung. Die Zeit war knapp, und ich musste am Abend Heim gehen, um an der VA für die Schule zu arbeiten. | | | |
| Zeitplanung | | | |
| Ich konnte mich nicht ganz an die zeitplanung einhaltung, denn ich habe einen Task welcher für den heutigen Tag geplant worden ist, nicht fertig machen können. | | | |
| Beanspruchte Hilfestellung | | | |
| Heute habe ich keine Hilfe in anspruch genommen. | | | |
| Nächste Schritte | | | |
| Ich werde mich gleich damit befassen, den Projektantrag so schnell wie möglich zu machen. Danach werde ich mir die restlichen Tasks vom zweiten Tag vornhemen. | | | |

Tabelle 2: Arbeitsjournal Tag 1

|  |  |  |  |
| --- | --- | --- | --- |
|  | Tag 2 22.11.2018 | |  |
| Soll | | Ist | |
| Projektantrag zusammenfassen | | Erledigt | |
| Soll/Ist Analyse | | Erledigt | |
| Use Cases erstellen | | Erledigt | |
| Software Architektur erstellen | | Erledigt | |
| ERD Design | | Fast erledigt | |
| Positives (Was lief gut?) | | | |
| Heute habe ich mich wieder hauptsächlich mit dem Dokumentieren befasst. Dabei habe ich mir als Ziel gesetzt, das I und P vom IPERKA fertigzustellen. Ich musste mich sehr spurten um das zu erreichenb, doch ich habe alle ausser einen Task fertig abschliessen können. | | | |
| Negatives (Wo gab es Probleme?) | | | |
| Ich bin nicht ganz fertig geworden mit dem ERD Design. Das lag daran, dass ich am Morgen noch den Projektantrag verfassen musste. | | | |
| Zeitplanung | | | |
| Ich konnte mich gut an die Zeitplanung halten. Das ERD ist fast fertig, und ich musste noch eine Arbeit vom vorherigen Tag erledigen. Jedoch liege ich trozdem bei der Arbeit zurück. | | | |
| Beanspruchte Hilfestellung | | | |
| Heute habe ich keine Hilfe benötigt. | | | |
| Nächste Schritte | | | |
| Als nächstes werde ich das ERD fertigstellen und das Testprotokoll schreiben. Zudem werde ich die Entscheidung treffen und diese Dokumentieren. Die ersten drei Schritte des IPERKAs (IPE) sollten fertig sein. Erst dann werde ich damit beginnen das Programm zu realisieren. | | | |

Tabelle 3: Arbeitsjournal Tag 2

|  |  |  |  |
| --- | --- | --- | --- |
|  | Tag 3 23.11.2018 | |  |
| Soll | | Ist | |
| ERD Design fertigstellen | | Erledigt | |
| Testprotokoll vorbereiten | | Erledigt | |
| Entscheidung dokumentieren | | Erledigt | |
| Frontend erstellen | | Erledigt | |
| Routing für alle Views | | Erledigt | |
| Positives (Was lief gut?) | | | |
| Heute konnte ich gut aufholen. Obwohl ich noch einen Task vom letzten Tag erledigen musste, sind alle Aufgaben, welche ich für diesen Tag geplant habe fertig geworden. | | | |
| Negatives (Wo gab es Probleme?) | | | |
| Heute hatte ein paar unklarheiten als ich das Frontend erstellt habe, doch ich konnte meine Probleme schnell lösen. | | | |
| Zeitplanung | | | |
| Ich konnte mich an die heutige Zeitplanung halten, denn ich halbe alles was geplant worden ist erledigt. Zudem habe ich auch eine Aufgabe vom vorherigen tag gemacht. | | | |
| Beanspruchte Hilfestellung | | | |
| Ich musste manchmal im Internet nachschauen, wie man einige Elemente im Frontend am besten erstellt. | | | |
| Nächste Schritte | | | |
| Als nächstes werde ich versuchen die Implementation des ganzen Programmes zu erstellen. Dazu gehören die verschiedenen funktionen des Spiels und die Highscoreliste. Ich hoffen das ich dies fertig schaffen werden. | | | |

Tabelle 4: Arbeitsjournal Tag 3

|  |  |  |  |
| --- | --- | --- | --- |
|  | Tag 4 27.11.2018 | |  |
| Soll | | Ist | |
|  | |  | |
|  | |  | |
|  | |  | |
|  | |  | |
|  | |  | |
|  | |  | |
|  | |  | |
| Positives (Was lief gut?) | | | |
|  | | | |
| Negatives (Wo gab es Probleme?) | | | |
|  | | | |
| Zeitplanung | | | |
|  | | | |
| Beanspruchte Hilfestellung | | | |
|  | | | |
| Nächste Schritte | | | |
|  | | | |

Tabelle 5: Arbeitsjournal Tag 4

|  |  |  |  |
| --- | --- | --- | --- |
|  | Tag 5 (1/2 Tag) 28.11.2018 | |  |
| Soll | | Ist | |
|  | |  | |
|  | |  | |
|  | |  | |
|  | |  | |
|  | |  | |
|  | |  | |
|  | |  | |
| Positives (Was lief gut?) | | | |
|  | | | |
| Negatives (Wo gab es Probleme?) | | | |
|  | | | |
| Zeitplanung | | | |
|  | | | |
| Beanspruchte Hilfestellung | | | |
|  | | | |
| Nächste Schritte | | | |
|  | | | |

Tabelle 6: Arbeitsjounal Tag 5

# Teil 2: Projekt-Dokumentation

## Informieren

### Soll-Analyse

Bei der Soll-Analyse werden die Anforderungen genau analysiert.

#### GUI

Das GUI wird in diesem Projekt mit Razor und CSS erstellt. Bei Bedarf werden einige jQuery Elemente im Frontend vorhanden sein.

Wenn die Applikation geöffnet wird, kommt man auf die Startseite eines ASP.NET Projektes. Es wurden keine Änderungen oder Vorarbeiten geleistet.

Standardmässig hat es jedoch ein gutes Grundgerüst, nämlich eine Navigation mit einer Startseite, Infoseite, Kontaktseite und eine Getting started Seite. Die getting started Seite wird bearbeitet, damit diese dann zur starseite meines Projektes wird.

Das Design des GUIs ist sehr schlicht gehalten, deswegen wird es noch einiges zum Verändern geben. Zudem sollte die Seite so Benutzerfreundlich wie möglich werden.

#### Ist-Analyse

Der Zweck der Ist-Analyse ist, herauszufinden was das vorhandene System bereits zu bieten hat.

**Hardware**

Das Entwicklungsgerät ist mein Geschäfts PC, und zwar einen Dell Precision M6700, mit dem Windows 10 Betriebssystem.

**Entwicklungsumgebung**

Das Projekt wird mit Visual Studio 2017 Enterprise realisiert. In dieser Edition ist ein ReSharper vorhanden. Alles ist schon konfiguriert für diese Implementation.

**SQL**

Mithilfe von Visual Studio und Microsoft MySQL Server Management Studio wird die Datenbank zum Laufen gebracht.

**Microsoft SQL Server Management Studio**

SQL Server Management Studio wurde installiert, damit man das im Visual Studio erstellte edmx ganz einfach importieren kann, und die Daten in die Datenbank einlesen kann.

**Internetbrowser**

Auf dem Gerät sind die Browser Chrome, Vivaldi, Edge und Internet Explorer installiert.

**Git**

Git ist schon seit langer Zeit auf dem Entwicklungsgerät installiert und ist bereit genutzt zu werden. Um die Arbeit mit Git zu erleichern, ist auch TortoiseGit installiert, damit Commits und Pushs schneller erledigt werden können.

### Use Cases

Um das Programm besser aufzeigen zu können, wurden Use-Cases erstellt.

**Spielzug machen**

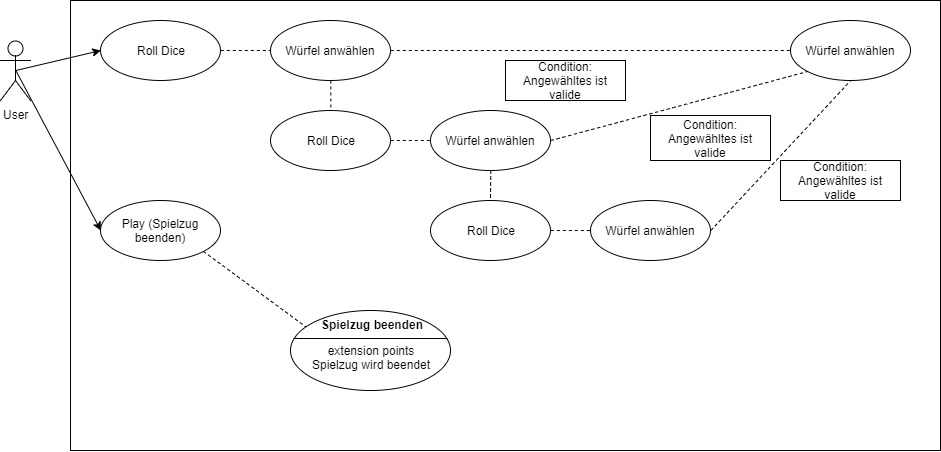


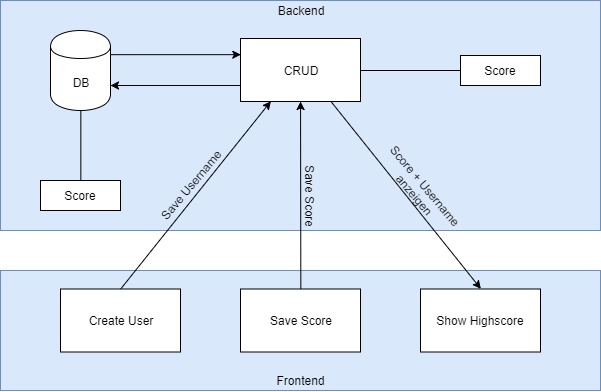
Abbildung 4: Spielzug Use Case

|  |  |
| --- | --- |
| Use-Case 1: Spielzug machen | |
| Beschreibung | Ein Spieler muss einen ganzen Spielzug durchführen können |
| Personen | User |
| Eintrittsfall | Der User ist an auf dem Spielbrett und ist gerade dran mit spielen. |
| Vorbedingungen | - |
| Resultat | Der User kann je nachdem wie der Spielzug verläuft einen machen, und die Würfel wählen welche er nach jedem Wurf behalten möchte. |
| Ablauf | |  |  |  | | --- | --- | --- | | Schritt | Person | Beschreibung | | 1 | User | User öffnet die Login Page | |

## Planen

### Software Architektur

Die Software Architektur beschreibt, wie der technische Aufbau der Applikation ist. Auf einer Grafik wurde dargestellt, wie das Projekt schlussendlich aufgebaut werden sollte.



Die Architektur ist folgendermassen aufgebaut:  
Die Applikation hat eine Datenbank (oben links) welche die Tabellen User und Item haben. Diese wird per Web.API an den Client gesendet. Die Web.API enthält alle CRUD Funktionen, damit die Daten vielseitiger genutzt werden können. CRUD steht für folgende Funktionen: Create, Read, Update und Delete. Für dieses Projekt sind zwar nicht alle notwendig, doch es werden trotzdem alle Funktionen implementiert, weil es sonst nicht CRUD wäre. Das CRUD verbindet beide, die User und die Item Tabelle mit dem Frontend.

Im Frontend sind drei verschiedene Aktionen welche die Applikation ausführen muss. Alle drei Aktionen sind mit dem CRUD verknüpft, damit die Daten mit der Datenbank ausgetauscht werden können.

**Create User**: Wenn ein User erstellt wird muss dieser in der Datenbank gespeichert werden.

**Save Score**:zu jedem User gehört ein Score. Dieser Score muss im gleichen Datensatz wie der Username gespeichert werden. Die Scores sollten nach grösse sortiert sein.

**Show Highscore**: Wenn der Benutzer auf die Highscore List geht, sollte er die fünf besten Scores welche gemacht wurden sehen. Die Liste sollte automatisch aktualisiert werden.

### Datenbankdesign

Die Datenbank für dieses Projekt ist klein. Sie besteht aus lediglich einer Tabelle, in der die Namen und Scores der Player gespeichert wurden. Da es nur eine Tabelle hat, hat es auch keinerlei Beziehungen.

Die Datenbank wird mithilfe von Visual Studio und SQL Server Management Studio realisiert.

###### Entity-Relationship-Model

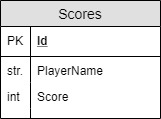


Abbildung 5: ERD

Um die Applikation noch performanter zu machen, wird es für einige Datensätze eine limitierte Zeichenlänge für die Strings haben. Auf folgender Tabelle ist ersichtlich, wie die Properties eingestellt werden sollten.

**Scores**

|  |  |  |
| --- | --- | --- |
| Typ | Name | Kommentar |
| PK | Id |  |
| String | PlayerName | Max. Length: 255, Nullable: False |
| Int | Score | Nullable: False |

Tabelle 7: Score ERD Anforderungen

###### Tabellen

###### Score

In dieser Tabelle sind alle Scores der Spieler gespeichert.

|  |  |
| --- | --- |
| Attribut | Beschreibung |
| Id | Jede Tabelle braucht eine Id, damit alle Datensätze Unique sind. |
| PlayerName | Immer wenn ein neues Spiel begonnen wird, muss der Player einen neuen Namen wählen, damit dieser sich schnell auf der Highscoreliste finden kann. (Falls er zu den Top 5 gehört.) |
| Score | Der Score wird von jedem Spieler in der Datenbank gespeichert. Die ersten fünf werden auf der Highscoreliste angezeigt sein. |

### Tesprotokoll

Nach der Implementierung des Projektes, muss man die Applikation auf Herz und Nieren prüfen. Vor dem Testen wurden Testfälle verfasst. Unten sieht man die Protokolle dieser Tests und das Resultat.

|  |  |
| --- | --- |
| Testfall 1: Buttons (Navigation) funktioniert | |
| Vorbedingungen: | Alle Buttons in der Aufgabenstellung sollten i GUI vorhanden sein. |
| Testablauf: | |  |  | | --- | --- | | Schritt | Beschreibung | | 1 | Klick auf einen beliebigen Button | | 2 | Weiterleitung auf die korrekte Page | |
| Testmittel: | Vivaldi Browser |
| Erwartetes Resultat | Die Buttons sollte den Benutzer bei Klick auf die Bullets auf die richtige Page leiten. Testablauf sollte bei allen Bullet Points ausgeführt werden und ein erfolgreiches Resultat aufweisen. |

|  |  |
| --- | --- |
| Testfall 2: Highscores werden gezeigt | |
| Vorbedingungen: | Die Liste sollte vorhanden und implementiert sein. Zudem sollten sich schon Ergebnisse in der Datenbank befinden. |
| Testablauf: | |  |  | | --- | --- | | Schritt | Beschreibung | | 1 | Klick auf den «Highscore» Button | | 2 | Liste wird geordnet angezeigt | |
| Testmittel: | Vivaldi Browser |
| Erwartetes Resultat | Die höchsten Scores sollten geordnet auf der Highscore Liste erscheinen. Wenn ein neuer Highscore entstanden ist, sollte dieser automatisch hinzugefügt werden können. |

|  |  |
| --- | --- |
| Testfall 3: Quit | |
| Vorbedingungen: | Der Quit Button auf der Startseite sollte vorhanden sein |
| Testablauf: | |  |  | | --- | --- | | Schritt | Beschreibung | | 1 | Klick auf den Quit Button | |
| Testmittel: | Vivaldi Browser |
| Erwartetes Resultat | Nach Klick auf den Button, sollte sich das Fenster schliessen. |

|  |  |
| --- | --- |
| Testfall 4: Einen Spielzug machen | |
| Vorbedingungen: | Das Spielbrett muss vollständig angezeigt und implementiert sein. Zudem müssen zwei Spieler am spielen sein. |
| Testablauf: | |  |  | | --- | --- | | Schritt | Beschreibung | | 1 | Klick auf «Play» | | 2 | Beim Spielbrett auf «Roll» klicken | | 3 | Würfel wählen welche man eintragen möchte | | 4 | Bei bedarf den 2. Und 3. Schritt wiederholfen | | 5 | Klick auf «Play» | |
| Testmittel: | Vivaldi Browser |
| Erwartetes Resultat | Über dem «Roll» Button befinden sich fünf Placeholder. Wenn man auf den Button geklickt hat, sollten in diesen Palceholdern eine random Anordnung von Würfeln zu sehen sein. Danach kann man die Würfel, welche man behlaten möchte eintragen. Diesen Vorgang bis zu drei Mal wiederholen.  Wenn der Klick auf «Play» geschehen ist, sollte der Zug des aktuellen Spielers beendet sein. |

|  |  |
| --- | --- |
| Testfall 4: Speziellen Spielzug landen | |
| Vorbedingungen: | Das Spielbrett muss vollständig angezeigt und implementiert sein. Zudem müssen zwei Spieler am spielen sein. |
| Testablauf: | |  |  | | --- | --- | | Schritt | Beschreibung | | 1 | Klick auf «Play» | | 2 | Beim Spielbrett auf «Roll» klicken | | 5 | Klick auf «Play» | |
| Testmittel: | Vivaldi Browser |
| Erwartetes Resultat | Über dem «Roll» Button befinden sich fünf Placeholder. Wenn man auf den Button geklickt hat, sollten in diesen Palceholdern eine random Anordnung von Würfeln zu sehen sein. Danach kann man die Würfel, welche man behlaten möchte eintragen. Diesen Vorgang bis zu drei Mal wiederholen.  Wenn der Klick auf «Play» geschehen ist, sollte der Zug des aktuellen Spielers beendet sein. |

## Entscheiden

## Realisieren

## Kontrollieren

## Auswerten

### Reflexion

### Optimierungsbedarf

### Schlusswort

# Teil 3: Anhang

## Quellenverzeichnis

|  |  |
| --- | --- |
| Was? | Quelle |
|  |  |

Tabelle 8: Quellenverzeichnis

## Abbildungsverzeichnis

[Abbildung 1: Scoring Zahlen einzeln 7](#_Toc531092876)

[Abbildung 2: Scoring Zahlen Spezialfälle 8](#_Toc531092877)

[Abbildung 3: IPERKA 11](#_Toc531092878)

[Abbildung 4: ERD 20](#_Toc531092879)

## Tabellenverzeichnis

[Tabelle 1: Meilensteine 13](#_Toc530643976)

[Tabelle 2: Arbeitsjournal Tag 1 15](#_Toc530643977)

[Tabelle 3: Arbeitsjournal Tag 2 16](#_Toc530643978)

[Tabelle 4: Arbeitsjournal Tag 3 17](#_Toc530643979)

[Tabelle 5: Arbeitsjournal Tag 4 18](#_Toc530643980)

[Tabelle 6: Arbeitsjounal Tag 5 19](#_Toc530643981)

## Glossar

|  |  |
| --- | --- |
| Begriff | Erklärung |
| Glossar | Im Glossar werden Begriffe eingefügt, welche erklärt oder übersetzt werden. |

Tabelle 9: Glossar

## Programmcode

Hier ist der Programmcode welcher in den fünf Tagen Projektzeit geschrieben wurde. Der Code wurde nach den verschiedenen Projekten getrennt.

### Yahtzy.Web

#### Controllers

#### Views

##### Game

**Index.cshtml**

<h2>Please enter the names of the players and click "start" to begin the game.</h2>

<label>Player 1</label>

<input type="text" />

<br />

<label>Player 2</label>

<input type="text" />

<br/>

<button type="button">@Html.ActionLink("Cancel", "Index", "Start")</button><br />

<button type="button">@Html.ActionLink("Start", "Game", "Game")</button><br />

**Game.cshtml**

<h2>Game</h2>

<div class="gameBlockLeft">

<div>

<table border="1">

<tr>

<th></th>

<th></th>

<th>Player 1</th>

<th>Player 2</th>

</tr>

<tr>

<th>Aces</th>

<th><img src="../../img/dice-1-md.png" style="width: 50px;"/></th>

<th></th>

<th><input type="text" class="boxSm" /></th>

</tr>

<tr>

<th>Twos</th>

<th><img src="../../img/dice-2-md.png" style="width: 50px;" /></th>

<th></th>

<th><input type="text" class="boxSm" /></th>

</tr>

<tr>

<th>Threes</th>

<th><img src="../../img/dice-3-md.png" style="width: 50px;" /></th>

<th></th>

<th><input type="text" class="boxSm" /></th>

</tr>

<tr>

<th>Fours</th>

<th><img src="../../img/dice-4-md.png" style="width: 50px;" /></th>

<th></th>

<th><input type="text" class="boxSm" /></th>

</tr>

<tr>

<th>Fives</th>

<th><img src="../../img/dice-5-md.png" style="width: 50px;" /></th>

<th></th>

<th><input type="text" class="boxSm" /></th>

</tr>

<tr>

<th>Sixes</th>

<th><img src="../../img/dice-6-md.png" style="width: 50px;" /></th>

<th></th>

<th><input type="text" class="boxSm" /></th>

</tr>

</table>

</div>

<div>

Its your turn, Player! <br />

<div class="boxDice"></div>

<div class="boxDice"></div>

<div class="boxDice"></div>

<div class="boxDice"></div>

<div class="boxDice"></div>

<br />

<button type="button">Roll</button>

<button type="button">Play</button>

</div>

</div>

<div class="gameBlockRight">

<div>

<table border="1">

<tr>

<th></th>

<th>Player 1</th>

<th>Player 2</th>

</tr>

<tr style="height: 50px">

<th>Three of a kind</th>

<th></th>

<th><input type="text" class="boxSm" /></th>

</tr>

<tr style="height: 50px">

<th>Four of a kind</th>

<th></th>

<th><input type="text" class="boxSm"/></th>

</tr>

<tr style="height: 50px">

<th>Full house</th>

<th></th>

<th><input type="text" class="boxSm" /></th>

</tr>

<tr style="height: 50px">

<th>Small Straight</th>

<th></th>

<th><input type="text" class="boxSm" /></th>

</tr>

<tr style="height: 50px">

<th>Large Straight</th>

<th></th>

<th><input type="text" class="boxSm" /></th>

</tr>

<tr style="height: 50px">

<th>Yahtzee</th>

<th></th>

<th><input type="text" class="boxSm" /></th>

</tr>

<tr style="height: 50px">

<th>Chance</th>

<th></th>

<th><input type="text" class="boxSm" /></th>

</tr>

</table>

</div>

<div>

<h3>Total Score</h3> <br />

Player 1 <br/>

Player 2

</div>

</div>

##### Highscore

**Index.cshtml**

<h2>Highscore</h2>

<table border="1">

<tr>

<th>Place</th>

<th>Name</th>

<th>Score</th>

</tr>

<tr>

<th>PlaceNumber</th>

<th>NamePlayer</th>

<th>ScorePlayer</th>

</tr>

</table>

<button type="button">@Html.ActionLink("Cancel", "Index", "Start")</button>

##### Shared

**\_Layout.cshtml**

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8" />

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Yahtzi</title>

@Styles.Render("~/Content/css")

@Scripts.Render("~/bundles/modernizr")

</head>

<body>

<div class="container body-content">

@RenderBody()

<hr />

</div>

@Scripts.Render("~/bundles/jquery")

@Scripts.Render("~/bundles/bootstrap")

@RenderSection("scripts", required: false)

</body>

</html>

**Error.cshtml**

<!DOCTYPE html>

<html>

<head>

<meta name="viewport" content="width=device-width" />

<title>Error</title>

</head>

<body>

<hgroup>

<h1>Error.</h1>

<h2>An error occurred while processing your request.</h2>

</hgroup>

</body>

</html>

##### Start

**Index.cshtml**

<h1>Yatzy</h1>

<button type="button">@Html.ActionLink("Play", "Index", "Game")</button><br />

<button type="button">@Html.ActionLink("Highscore", "Index", "Highscore")</button><br />

<button type="button" onclick="self.close();">Quit</button><br />

#### Models

### Yahtzy.Domain

**YahtzyModel.edmx.sql**

-- --------------------------------------------------

-- Entity Designer DDL Script for SQL Server 2005, 2008, 2012 and Azure

-- --------------------------------------------------

-- Date Created: 11/27/2018 14:03:18

-- Generated from EDMX file: C:\Git\Yahtzi\src\Yahtzy.Domain\Model\YahtzyModel.edmx

-- --------------------------------------------------

SET QUOTED\_IDENTIFIER OFF;

GO

USE [Yahtzy];

GO

IF SCHEMA\_ID(N'dbo') IS NULL EXECUTE(N'CREATE SCHEMA [dbo]');

GO

-- --------------------------------------------------

-- Dropping existing FOREIGN KEY constraints

-- --------------------------------------------------

-- --------------------------------------------------

-- Dropping existing tables

-- --------------------------------------------------

-- --------------------------------------------------

-- Creating all tables

-- --------------------------------------------------

-- Creating table 'Scores'

CREATE TABLE [dbo].[Scores] (

[Id] int IDENTITY(1,1) NOT NULL,

[PlayerName] nvarchar(255) NOT NULL,

[Score] bigint NOT NULL

);

GO

-- --------------------------------------------------

-- Creating all PRIMARY KEY constraints

-- --------------------------------------------------

-- Creating primary key on [Id] in table 'Scores'

ALTER TABLE [dbo].[Scores]

ADD CONSTRAINT [PK\_Scores]

PRIMARY KEY CLUSTERED ([Id] ASC);

GO

-- --------------------------------------------------

-- Creating all FOREIGN KEY constraints

-- --------------------------------------------------

-- --------------------------------------------------

-- Script has ended

-- --------------------------------------------------

### Yahtzy.BusinessLogic