

# ÍSLENSKA HUG BÚNAÐAR FABRIK KAN 2017

KYNNIR



Arnar Freyr Sævarsson  
Barði Freyr Þorsteinsson  
Valdimar Jónsson

## TABLE OF CONTENTS

<b>1. Project Overview</b>	<b>4</b>
1.1. Description	4
1.2. System Overview	5
1.2.1. Overview	5
1.2.2 Database Schema	6
1.2.3. General UI Wireframe	7
1.2.4. Admin UI Wireframe	8
1.3. Definitions	10
1.4. User Groups	11
1.5. User Tests	12
1.5.1. Notandapróf (In Icelandic)	12
1.5.2. Niðurstaða	19
<b>2. Risk Assessment</b>	<b>20</b>
2.1. Assessment Analysis	20
2.2. Risk Analysis	20
2.3 Risks that came up	22
<b>3. Working Agreement</b>	<b>22</b>
3.1. Rules of engagement	22
3.2. Coding Rules	23
3.2.1. C#	23
3.2.2. Javascript/Typescript	23
3.2.3. General	24
3.3. Sprint Estimates	24
3.4 Scrum	27
<b>4. Tech Overview</b>	<b>28</b>
4.1. Tech	28
4.2. Build pipeline and Deployment	29
4.2.1. Jenkins	29
4.2.2. AWS	29
4.2.3. Ansible, deployment and hosting	29
4.2.4 Overview of Build pipeline	31
4.2.4.1 Build pipeline for branches	32
4.2.4.2 Build pipeline for master	34
4.3. Testing	35
4.3.1. Unit testing	35
4.3.1.1 General UI tests	35
4.3.1.2 Admin tests	35
4.3.1.3 API tests	36

---

<b>5. Progress Report</b>	<b>37</b>
5.1. Sprint 0	37
5.1.1. Working draft	37
5.1.2. Retrospective	37
5.2. Sprint 1	38
5.2.1. Burndown Chart	38
5.2.2. Retrospective	38
5.3. Sprint 2	39
5.3.1. Burndown Chart	39
5.3.2. Retrospective	39
5.4. Sprint 3	40
5.4.1. Burndown Chart	40
5.4.2. Retrospective	40
5.5. Sprint 4	41
5.5.1. Burndown Chart	41
5.5.2. Retrospective	41
5.6. Sprint 5	42
5.6.1. Burndown Chart	42
5.6.2. Retrospective	42
5.7. Sprint 6	43
5.7.1. Burndown Chart	43
5.7.2. Retrospective	43
5.8 Sprint 7	44
5.8.1. Burndown Chart	44
5.8.2. Retrospective	44
5.9 Sprint 8	45
5.9.1. Burndown Chart	45
5.9.2. Retrospective	45
5.10 Overview of Sprints	46
5.10.1. Finished requirements over time	46
5.10.2 Time spent on assignment	46
<b>6. Requirements</b>	<b>47</b>
6.1. Explanation	47
6.2. Functional Requirements	47
6.2 Non-functional requirements	51
<b>7. Diary</b>	<b>52</b>

---

## 1. Project Overview

### 1.1. Description

Símamótið was previously a website with a focus on mobile users, to help the organizers and viewers of the football tournament with the same name. That project is very unsuitable for improvements and is only capable of supporting one type of a football tournament structure, which is Símamótið. Our goals are to redesign the system, to generalize it to work for different types of tournament structures, and also make it more suitable for future improvements along with making it scalable so that the system can support multiple sport types. This means a complete redesign and overhaul of both the API and database, including both user interfaces.

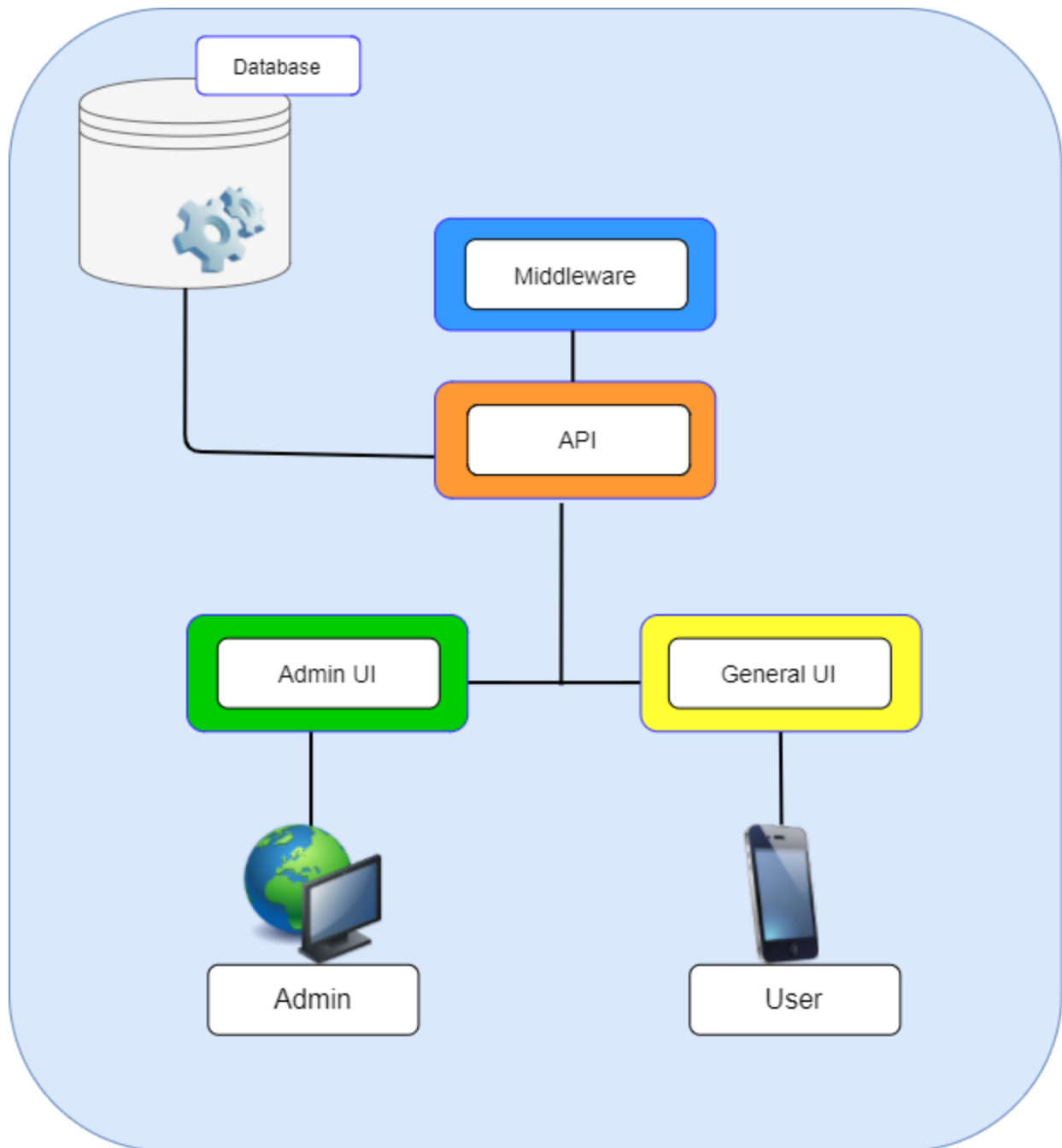
The general user interface will simply serve to display data to the user, with emphasis on being easy to use and simple, with rich text and images. The API will be a data manipulation REST API. It will grant higher access to logged in users, gather relevant data from database, and be a communicator to another factor in our system, the so called middleware. The middleware will take care of sport-specific data, and by doing so we simplify the base API, the database will mainly store non-specific data and the API will have generic functions for manipulating such data. The middleware will then take care of more specific tasks for each type of sporting event. An example of this would be, the database stores a previous match. The middleware determines the next match for the team, using the existing data from the database. This means we can exclude sport-specific rules from the base API functions, but the middleware can calculate/create any necessary data for a given situation. The purpose of this type of middleware is to simplify the addition of new types of sporting events.

With this system, we will be focusing on sports with a similar structure to football. That is, two teams competing, each with one or more players trying to outscore the opposing team. This means score-based winner results, but scores can be omitted in the system if scorekeeping is omitted as well, like in matches between very young players.

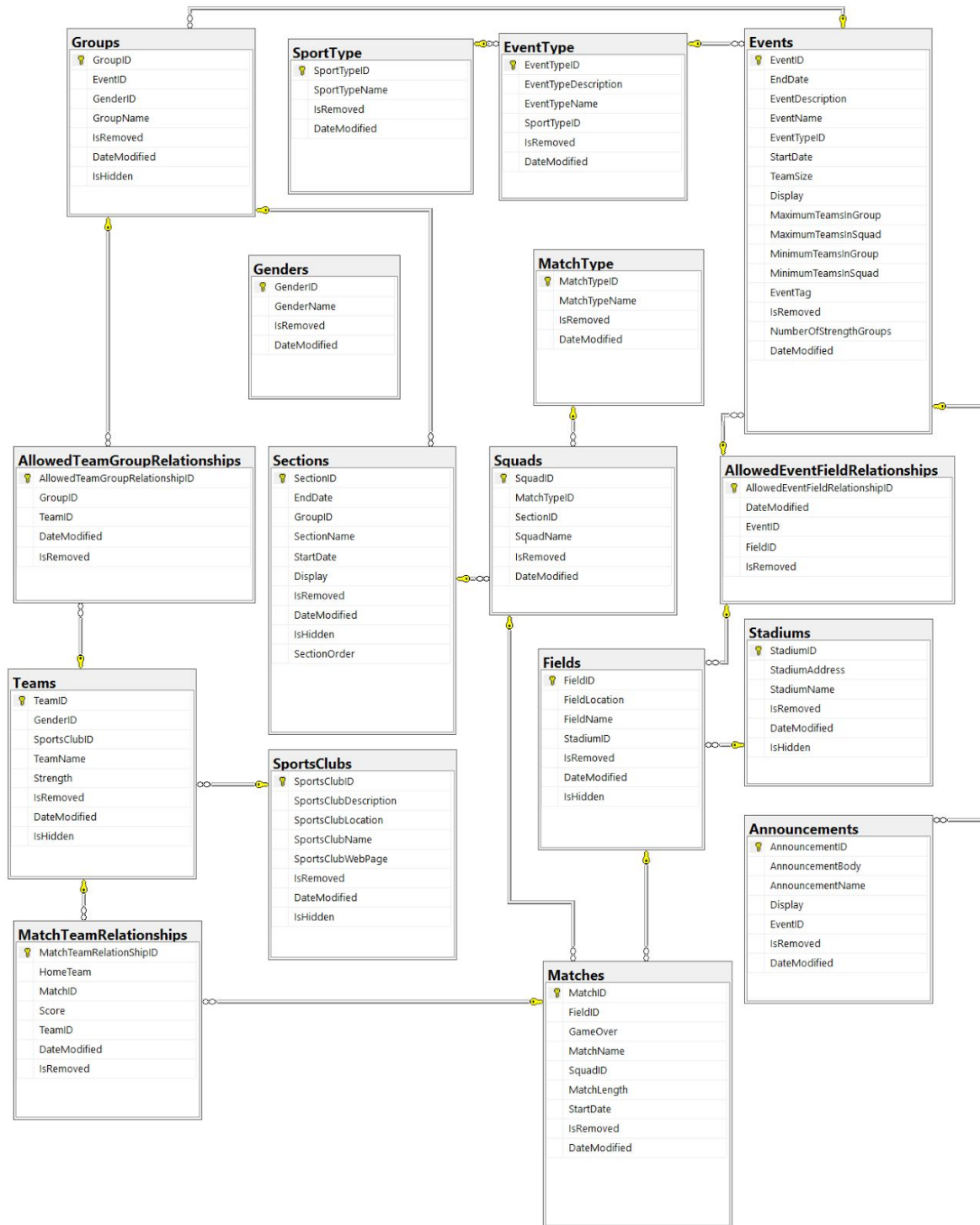
We decided to add another UI element which we call the Admin UI, or simply Admin. This is a completely separate website to the general UI. It will serve to create and manage events which will then be visible in the UI. Here an Admin type user will login, and in the case of MVP an event organizer in most cases, and be able to create an event as he chooses, and manipulate that data to suit the needs of the event he is creating.

## 1.2. System Overview

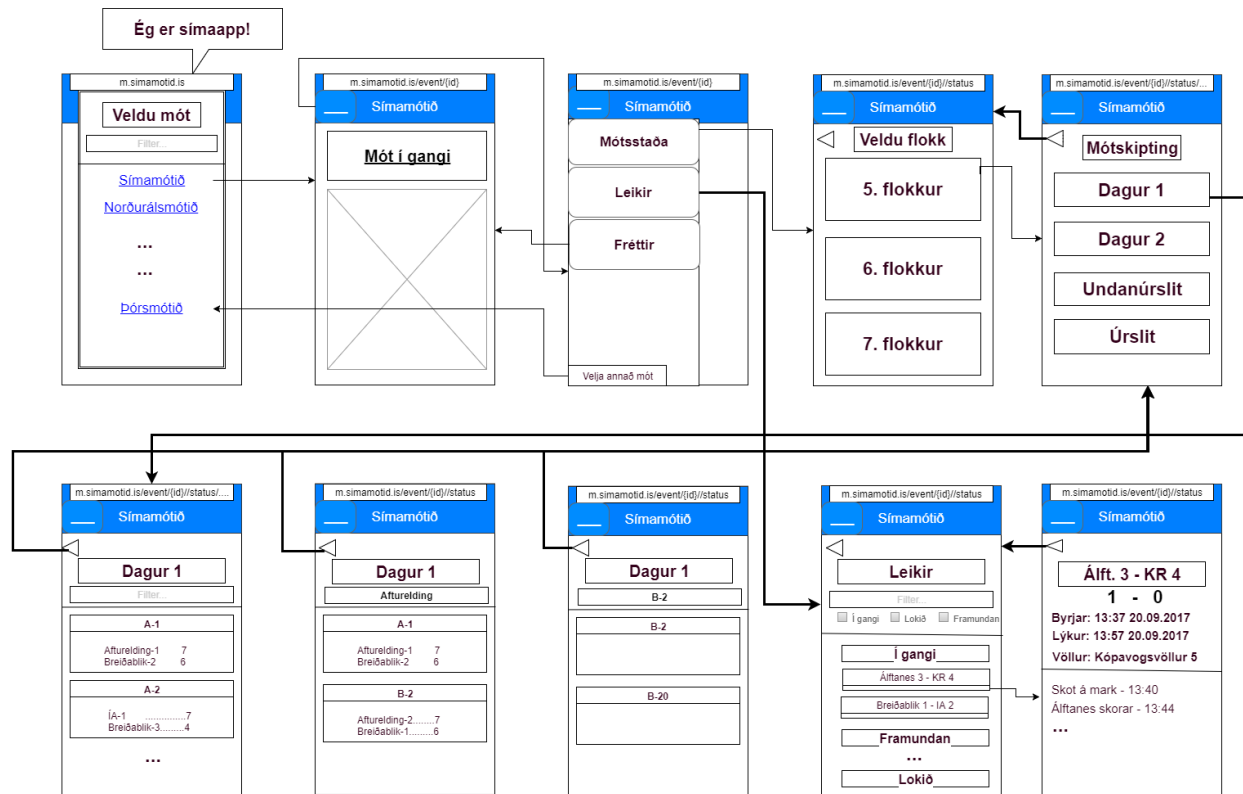
### 1.2.1. Overview



## 1.2.2 Database Schema



### 1.2.3. General UI Wireframe



## 1.2.4. Admin UI Wireframe



- 
- The concept draft of the admin system can be seen above. It was used as a center point for designing the admin user interface, of which screenshots are shown in greater detail below. This website was not designed to be mobile first, but rather as a web based application to be used on a laptop, tablet or a PC, contrary to the general UI.

### 1.3. Definitions

The following are all terms used throughout the report. They are defined here to avoid any confusion regarding their meaning.

- **Match:** When teams compete against one another in a particular sport. (ísl: leikur)
- **Event:** Takes place at a given date, in a particular sport at a certain location. Matches take place at an Event. (ísl: mót)
- **User:** The general user of this system. Can view every events and matches. No login required. (ísl: notandi)
- **Admin:** The administrative user of the system. Has access to every feature of the system. (ísl: stjórnandi)
- **Event Organizer:** The user manipulating event, match and team data. (ísl: skipuleggjandi móts)
- **Staff:** Intended for users that will be following matches, and registering the score, start/stop matches and add descriptions. (ísl: starfsfólk)
- **Referee:** Appointed to a match. As an extension to the staff, the referee can edit match data after they have been closed. (ísl: dómari)
- **Team:** Forms one side of players in a given sport. One or more players. Zero or more coaches. (ísl: lið)
- **Team Organizer:** Manages which players and coaches are in a team. (ísl: liðstjóri)
- **Started match:** A match can be started manually, or if score is changed it will automatically start. Once started, the starting time of the match will be finalized. (ísl: hafinn leikur)
- **Closed match:** A match can be closed at any given time, whether it has been started or not. If a match is closed without having been started, the start time of the match will be set to the expected start time. (ísl: lokaður leikur)
- **Team rating:** How viable the team is compared to other teams, the higher the better. (ísl: liðstyrkur)
- **Squad:** The teams that play at a certain stage against each other in the event form a squad. (ísl: riðill)
- **Groups:** This is a part of the event, more or less a sub-event in itself. This is necessary as groups of squads never play against each other, and might even have their own rules for progression within the event. (ísl: flokkur)
- **Sections:** Each event can have multiple different parts. Each part can have different start/end dates, algorithms (currently one available) and names. Sections are independant between each group (ísl: hluti)

## 1.4. User Groups

<b>User</b>	
A user can be anyone who uses the software. A user wants view information about specific events and matches, and get updates relevant to them.	
<b>Staff</b>	<b>Team Organizer</b>
Staff members are Users who can have many roles during events/matches. They can be assistant coaches, field employees, event employees and more. They can assist the referee in his duties.	Team organizers are Users who want to manage information about their team, and register their team for events/tournaments.
<b>Referee</b>	
A Referee is a staff member who wants to make sure the game is played according to the rules. S(he) wants to be able to assign scores to teams and players, and edit mistakes.	
<b>Event Organizer</b>	
An event organizer wants to be able to manage the entire event. S(he) wants to be able to assist and change information that staff and/or referees add. S(he) wants to be able to plan future events, as well as modify previous ones.	
<b>Admin</b>	
Admins have control over everything in the system. They can change data in the system.	

## 1.5. User Tests

### 1.5.1. Notandapróf (In Icelandic)

- Notandi - Tilvonandi notandi kerfis
- Verkefni - Búa til nýtt mót og bæta við viðeigandi upplýsingum.

 Viðmót Mótshaldara

### INNSKRÁNING

Notendanafn

arnarfs@siminn.is

Lykilorð

• • • • • |

Innskrá

Sækja um

Login virkaði án vandræða. Engin virkni er þó kominn í “Sækja um” flipann.



## Viðmót Mótshaldara

Velkominn

Nýtt mót

Leita eftir nafni móts

MÓT	ÍÐRÓTTATEGUND	MÓTSTEGUND	DAGSETNING	BIRTA
Símamótið	Fótbolti	Símamótið 7-manna	10 Nóv - 12 Des	<input checked="" type="checkbox"/>
Norðurlásmótið	Fótbolti	Símamótið 7-manna	8 Okt - 10 Okt	<input checked="" type="checkbox"/>
Cheerios mót Víkings	Handbolti	Cheerios mót Víkings	21 Des - 24 Des	<input checked="" type="checkbox"/>
Dominos Deildin	Körfubolti	Dominos Deildin	1 Jún - 6 Jún	<input checked="" type="checkbox"/>
Mótið	Fótbolti	Krakkamót 5-manna	11 Nóv - 12 Des	<input type="checkbox"/>
Name	Fótbolti	Símamótið 7-manna	11 Nóv - 12 Des	<input type="checkbox"/>
Name	Fótbolti	Símamótið 7-manna	11 Nóv - 12 Des	<input type="checkbox"/>

Notandi var mjög snöggur að finna út hvað hann átti að gera til að búa til nýtt mót.



## Viðmót Mótshaldara



IÓTSUPPLÝSINGAR



FLOKKAR



MÓTSHLUTAR



FJÖLDI



YFIRLIT



LOKID

## Símamót test PLG

#símamótttestplg2017

Nafn móts:

Símamót test PLG

Fótbolti -

Símamótið 7-manna -

Byrjar:

7 December 2017 08:30

Endar:

9 December 2017 15:00

Stutt lýsing

Prufa

Áfram

Ánægður með einfalt viðmót. Notandi vill geta skráð inn lengd á leikjum á þessari síðu.

Viðmót Mótshaldara

MÓTSUPPLÝSINGAR

FLOKKAR

MÓTSHLUTAR

FJÖLDI

YFIRLIT

LOKID

Símamót test PLG

FLOKKAR

Nafn flokks

6. Flokkur

Kvenna

Bæta við

Flokkar í móti

♀ 5. Flokkur

Til baka

Áfram

Notandi gat bætt við flokkum án vandræða, og hélt áfram á næstu síðu.

Viðmót Mótshaldara

MÓTSUPPLÝSINGAR

FLOKKAR

MÓTSHLUTAR

FJÖLDI

YFIRLIT

LOKID

Símamót test PLG

MÓTSHLUTAR

Nafn hluta

Úrslitakeppni - seinni hluti

Bæta við

Byrjar:

9 December 2017 14:30

Endar:

9 December 2017 15:00

Bæta hluta við eftirfarandi flokka:

♀ 5. Flokkur

♀ 6. Flokkur

Bæta við alla flokka

Hlutar

Styrkleikakeppni - dagur 1


Styrkleikakeppni - dagur 2








Úrslitakeppni - fyrrihluti

Til baka

Áfram

Notandi er ánægður með útlitið. Hann var hinsvegar aðeins ringlaður þegar hann var búinn að búa til þrjá hluta og var ekki viss hvenær hann lét seinasta enda og hvenær næsti ætti að byrja. Stungið upp á umbót þar sem næsti hluti fær byrjunardagsetningu eftir enda fyrri hluta.


 Viðmót Mótshaldara

Símamót test PLG  
 FJÖLDI

STYRKLEIKAFLOKKAR  
 Fjöldi styrkleikaflokka

1

FLOKKAR  
 Lágmarksfjöldi liða í flokki

80

Hámarksfjöldi liða í flokki

96

RIDLAR  
 Lágmarksfjöldi liða í riðli

10


Hámarksfjöldi liða í riðli








16

Til baka

Áfram

Notandi vill geta valið sérstaka styrkleikaflokka fyrir fyrir hvern flokk fyrir sig. Einnig vill hann geta sleppt því að skrá styrkleikaflokka á þessum tímapunkti.


 Viðmót Mótshaldara

Símamót test PLG  
 YFIRLIT

LÝSING  
 Prufa

TEGUND MÓTS  
 Íþróttategund: Fótbolli  
 Mótstegund: Símamótið 7-manna

DAGSETNING MÓTS  
 Byrjunardagsetning: 7 Desember 2017 kl: 08:30  
 Endadagsetning: 9 Desember 2017 kl: 15:00

FLOKKAR  
 ♀ 5. Flokkur  
 ♀ 6. Flokkur

MÓTSHLUTAR  
 Styrkleikakeppni - dagur 1:  
 Byrjunardagsetning: 7 Desember 2017 kl: 08:30  
 Endadagsetning: 9 Desember 2017 kl: 15:00

Notandi er ánægður með yfirlitssíðu. Vill geta smelt á einstaka hluti á yfirlitssíðunni til að geta breytt þeim.

## Viðmót Mótshaldara

### SÍMAMÓT TEST PLG

Tilkynningar Flokkar Hlutar Riðlar Lið Leikir Vellir Upplýsingar

Vill breyta röðun hér, Upplýsingar, Tilkynningar, Flokkar, Hlutar, Vellir, Lið, Bæta við Styrkleikaflokkar, Riðlar og loks Leikir

## Viðmót Mótshaldara



### SÍMAMÓT TEST PLG

Tilkynningar Flokkar Hlutar Riðlar Lið Leikir Vellir Upplýsingar

Nafn tilkynningar

Mótið hefst snemma

Bæta við

☒ Birta á forsiðu

B I U

Engar tilkynningar

Vista Hætta við

Fannst óljóst að þurfa að ýta bæði á “bæta við”, og “vista” til að skrá frétt á mótið. Vill geta gert frétt að “sticky” eða að hún birtist sjálfkrafa efst. Einnig að breyta röðun á fréttum.

## Viðmót Mótshaldara

### SÍMAMÓT TEST PLG

Tilkynningar **Flokkar** Hlutar Riðlar Lið Leikir Vellir Upplýsingar

FLOKKAR SKRÁÐIR Í MÓTI

Leit

♀ 5. FLOKKUR  
4 mótshlutar skráðir

♀ 6.FLOKKUR  
4 mótshlutar skráðir

SKRÁ FLOKK

Nafn

7. Flokkur

Kvenna

Hætta við Bæta við

Notandi gat bætt við flokkum án vandræða. Engar athugasemdir.

Viðmót Mótshaldara

Gott væri að sjá kyn á flokki á þessari síðu. Annars mjög fínt.

## SÍMAMÓT TEST PLG

TilkynningarFlokkarHlutarRiðlarLiðLeikirVellirUpplýsingar

### HLUTAR SKRÁÐIR Í MÓT

Leit

**5. FLOKKUR**

Styrkleikakeppni - dagur 1	x
Styrkleikakeppni - dagur 2	x
Úrslitakeppni - fyrrihluti	x
Úrslitakeppni - seinni hluti	x

**6. FLOKKUR**

Styrkleikakeppni - dagur 1	x
Styrkleikakeppni - dagur 2	x

### SKRÁ HLUTA

Nýr hluti

## Viðmót Mótshaldara

## SÍMAMÓT TEST PLG

Tilkynningar	Flokkar	Hlutar	Riðlar	<b>Lið</b>	Leikir	Vellir	Upplýsingar
--------------	---------	--------	--------	------------	--------	--------	-------------

### LIÐ SKRÁÐ Í MÓT

Leit

**BREIÐABLIK**

**FJÖLNIR**

5. flokkur: 11 lið skráð

**STYRLEIKAFLOKKUR 1**

Fjöldnir 1

Fjöldnir 2

**STYRLEIKAFLOKKUR 2**

**STYRLEIKAFLOKKUR 3**

Fjöldnir 3


Fjöldnir 4

**STYRLEIKAFLOKKUR 4**

### SKRÁ LIÐ

Veldu félag

Notandi finnst ekki þörf á því að birta tóma styrkleikaflokka. Hann vill geta endurskýrt lið á þessari síðu. Einnig vill hann bæta við nýjum flipa til að eiga betur við styrkleikaflokkana óháð liðum.

 Viðmót Mótshaldara

## SÍMAMÓT TEST PLG

Tilkynningar
Flokkar
Hlutar
Riðlar
Lið
Leikir
Vellir
Upplýsingar

### VELLIR SKRÁÐIR Í MÓT

Leit

---

**KÓRINN** ✓ + ✗

---

**VÖLLUR 1A** ✗

Staðsetning: Inni

### SKRÁ VELLI

Kórin

Fyrirfram skilgreindir vellir

Völlur 1 ✗

Nafn

Völlur 2b

Staðsetning

Úti

+

Hætta við

Bæta við

Notandi er mjög sáttur við þessa síðu, en þó vill hann geta bætt við mörgum völlum á sama tíma.

### 1.5.2. Niðurstaða

- Notandi er yfir heilt sáttur við viðmótið, og spenntur fyrir framhaldinu. Honum þótti viðmótið á köflum vera full “langt” og því erfitt að sjá heildarmyndina. Hann vill að við bætum við flipa fyrir styrkleikaflokka svo hægt sé að stilla þá sérstaklega. Hann vill geta tekið afrit af eldra móti til að flýta fyrir. Notanda var tjáð að útfærsla fyrir Riðla og Leiki var ekki tilbúið þegar notendaprófið átti sér stað, og því mun nýtt próf vera gert þegar sú útfærsla er tilbúin.

## 2. Risk Assessment

### 2.1. Assessment Analysis

Likelihood	Consequences				
	Insignificant	Minor	Moderate	Major	Catastrophic
Odds: >90%	High	High	Extreme	Extreme	Extreme
Odds: 50-90%	Moderate	High	Extreme	Extreme	Extreme
Odds: 10-50%	Low	Moderate	High	Extreme	Extreme
Odds: 3-10%	Low	Low	Moderate	High	Extreme
Odds: < 3%	Low	Low	Moderate	High	High

### 2.2. Risk Analysis

Risk	Conseq.	Odds	Analysis	Assessment	Responsible
No deliverable product	Catastrophic	<1%	If the team ends up with no deliverable product on the due date (18. Dec) the results are catastrophic. However since we are going to ensure Minimum Viable Product (MVP) by finishing requirements from start to finish(UI to database) to ensures that we always have something to deliver.	High	Arnar Freyr Sævarsson
Loss of code	Minor	5%	Losing code could lead to some serious problems if no precautions are taken. Git will be used for source control so the risk is minimised. This risk is mainly if someone doesn't add their code to source control and something happens to hardware/software in the	Low	Barði Freyr Þorsteinsson

			meantime. This means at most that code written that day is lost for that particular team member.		
Loss of data	Insignificant	5%	Loss of data in regards to the final assignment is not a serious issue. Since the project doesn't rely on gathered data, losing it doesn't have a major effect on the project.	Low	Barði Freyr Þorsteinsson
General UI server goes down	Minor	5%	If the AWS hosting the user interface goes down the results will be minor. This will lead to an inconvenience to users. Team members have taken necessary precautions to quickly establish a new running server instance	Low	Valdimar Jónsson
Admin server goes down	Minor	5%	If the AWS hosting the admin user interface goes down the results will be minor. This will lead to an inconvenience to administrators. Team members have taken necessary precautions to quickly establish a new running server instance	Low	Valdimar Jónsson
API server goes down	Minor	5%	If the AWS hosting the API goes down the results will be minor. This will lead to an inconvenience to users and administrators. Team members have taken necessary precautions to quickly establish a new running server instance	Low	Valdimar Jónsson
The scope of the project changes	Moderate	50%	The most likely scenario that could happen is that the project scope changes severely over the course of the semester. Precautions were taken in the form of rules specified to deliver a	Extreme	Arnar Freyr Sævarsson

			Minimum Viable Product		
--	--	--	------------------------	--	--

### 2.3 Risks that came up

During the duration of the project the only risk we encountered was that the project scope changed. The scope changed severely from the only requirement being that we develop an API, to developing the API and a general user interface, and then again to developing an admin user interface as well. The team dealt with this well and managed to finish all A requirements before the complete code freeze (13.12.2017).

The other risks analyzed above were not encountered during the project: The risks however were still minified in the sense that the time taken to recover is severely reduced. A good example of this would be if any of the three servers went down. The team created Ansible scripts with guidance from a Síminn employee, that could upon receiving new servers to host the websites/API on, completely initialize the server with all necessary requirements and launch the websites/API to production. This still results in some downtime but given that the websites/API are hosted on AWS servers we believe that the risk is minimal.

## 3. Working Agreement

### 3.1. Rules of engagement

- Working hours will be estimated **8 hours a day**, twice a week on Mondays and Thursdays. Working hours can be changed two days in advance, if need be.
- Daily standups will be conducted at least every business day even though no work has been done on the assignment.
- Daily standups shall be 10-15 minutes long and the status of the assignment discussed.
- Meetings with our connections at Síminn will be every 2 weeks and more frequently if needed.
- If a team member has spent 2 hours on a specific problem without results, he consults another team member.
- Reports and slides shall be written in **English**, but presentations can be in **Icelandic**.
- **Minimum Viable Product:** The project will be written such that features that are worked on will be completed or postponed before the next feature (Database-API-UI).
- **Master branch** shall always be clean and not be pushed directly to. Feature branches will be worked on and either merged with master once a feature is finished, or discarded.

- **Death** Penalties will be given in the form of a beer fine, added to “workplaces” beer tub.
- **3-Week period addition:** In this time period, we decided it would be best to maximize our efforts with no distractions from other courses, and work full-time, on weekends if necessary to deliver the project.

## 3.2. Coding Rules

### 3.2.1. C#

- Function Names and Class Names are declared in Pascal Casing:

```
public void FunctionName(int localVar)
```

- Local variables inside functions shall begin with an underscore:
- Non-Model(Entity) Variables shall be in Camel Casing:
- The first curly bracket is declared in a newline after a function/class declaration
- The second curly bracket is declared in a newline after the last line of code within the function/class

```
public void FunctionName(int localVar)
{
    int functionName = 4
    int _localVar = localVar;
}
```

- Comments should be in XML format (summary). Specific care shall be given to the explanation of return values.

```
/// <summary>
/// This function takes an input and meticulously crunches it.
/// </summary>
/// <param name="inputValue">Should be the number 42</param>
/// <returns>Possibly The answer to life, the universe and everything</returns>
```

### 3.2.2. Javascript/Typescript

- Identifier names shall be in Camel Casing (function and variable names).
- The first curly bracket is defined in the same line as the function with a space between the function declaration.
- The second curly bracket is in a newline after the last line of code within the function.

```
function functionName(variableName) {
    var name = 3;
}
```

- Always end simple statements with a semicolon.

```
function functionName() {
  var simpleStatement = 3;
  var stillSimple = {
    a = 3,
    b = 2,
    c = 1
  };
}
```

### 3.2.3. General

- Spaces put around operators (+, -, /, \*, =) and commas.

```
var x = 3 + 2;
var list = ["this", "is", "a", "list"];
```

- Try to use descriptive variable and function names.

```
public int getGameInfoById(int gameId)
{
  var gameInfo = _service.getGameInfo(gameId);
  return gameInfo;
}
```

- TSLint is used to validate that coding rules are followed in Typescript

### 3.3. Sprint Estimates

Sprint no.	Goals	Start date	Finish date	Points estimated / points assigned
0.	Design of the system....	30.08.2017	11.09.2017	0
1.	-Database analysis -UI design -API resource analysis -Setup of UI, Middleware, API and database -Setup of the build pipeline -Specification freeze at end of sprint (unless something extremely important comes up)	11.09.2017	25.09.2017	96 / 80
2.	CD for .net Core (8) (SCRAPPED) CD for Angular (6) (SCRAPPED) Create Entity framework Schema (5) Connect API to MSSQL Server (5) API resource analysis (10) Create a controller for: <ul style="list-style-type: none"> <li>• Events (6)</li> </ul>	25.09.2017	09.10.2017	96 / 92

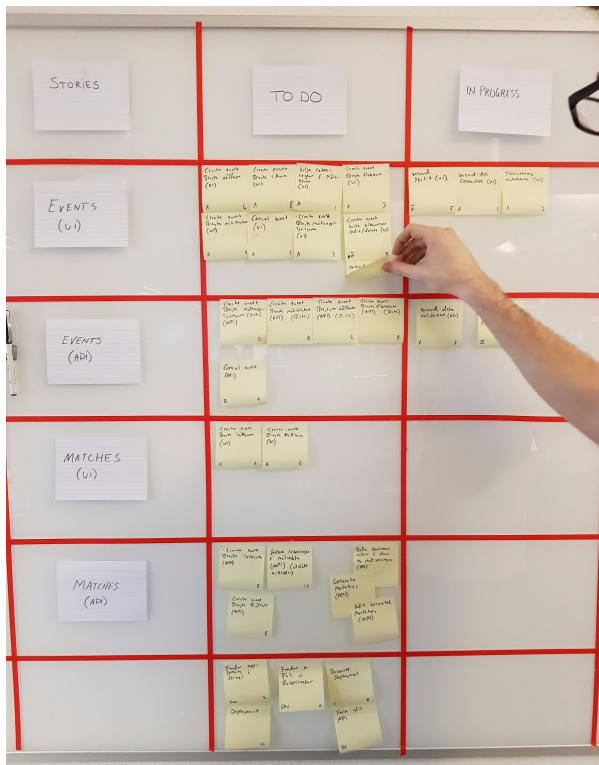
	<ul style="list-style-type: none"> <li>• Groups (5)</li> <li>• Matches (6)</li> <li>• Sections (6)</li> <li>• Squads (6)</li> </ul> View Information about sporting events (17) View Status of squads in an event (4) View groups for a event that interests me (4) View Sections for the events that interest me (4)			
3.	Minor finishing touches to previous sprint (8) Evaluate current specification / assigned points (12) Create Event from wizard info (36) - <b>(NOT FINISHED)</b> Filter (20) -(Complete) Specification freeze at end of sprint - In this sprint we realised that the times taken were significantly underestimated and requirements redone	09.10.2017	23.10.2017	96 / 76
4.	Edit/delete on Event information (56) <b>(NOT FINISHED)</b> Create Event from wizard info (36) - <b>(Finishing touches)</b> Set up authentication (10) Wizard-API connection (4)	23.10.2017	06.11.2017	96 / 80
5.	-Exam preparations, little work completed/assigned Create a controller for Announcements (6) Create another admin (3)	06.11.2017	27.11.2017	24 / 9
6.	Edit/delete on Event information (56) <b>(Finished)</b> Edit Groups of teams (10) Remove Groups of teams (10) Change teams participating in event (14) Create a field (7) Change stadiums/fields in an event (8) Edit event section timing (6) Change Groups within an event (11) Change event sections (12)	27.11.2017	04.12.2017	120 / 92

7.	Edit the score of a match (5) Automatically have squads set up for given section (36) Edit automatically generated squads (10) Remove entire squads (8) Choose an algorithm for each event section (17) Cancel my event/s). (7) Deploy the Websites on a server with a domain name (20) -Feature freeze at the end of sprint	04.12.2017	11.12.2017	120 / 108
8.	-Finishing touches on unfinished requirements that were previously started (within reason) -Bug fixing -Code freeze at least three days before presentation -Final presentation work	11.12.2017	18.12.2017	120 / 79

### 3.4 Scrum

Team members followed a simple scrum process while developing the software. As mentioned in the Rules of Engagement chapter above, team members took daily standups on each day that took approximately 10-15 minutes. In that period of time the team went up to their physical scrum board and discussed tasks that were completed or not completed during the previous day and discussed what tasks are to be assigned that day. The scrum board at the start of the code freeze is shown on the right with the only tasks remaining related to report and presentation work

28.11.2017



14.12.2017



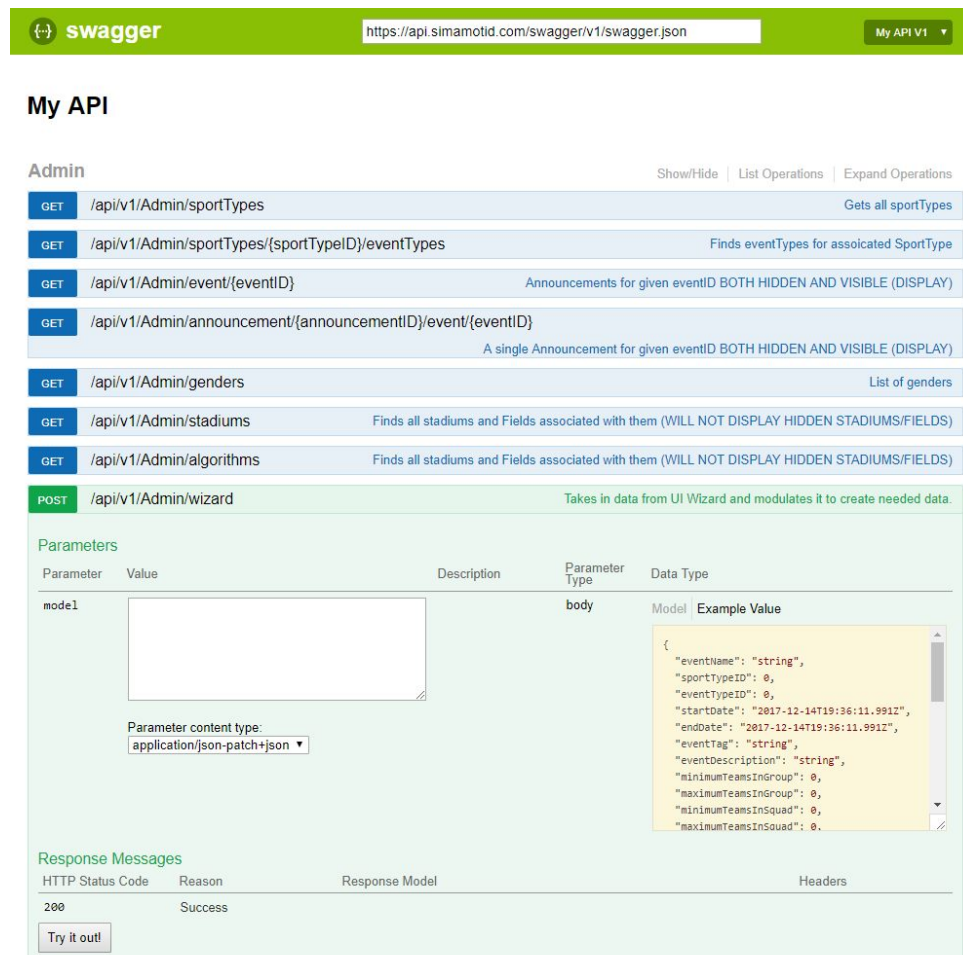
## 4. Tech Overview

### 4.1. Tech

On the software side of things, for the base API, ASP.NET Core 2.0 is used and by extension C#, with Visual Studio 2017 as IDE. For the front end, we are using Angular 4 with Visual Studio Code as IDE for it's nice support for typescript. For continuous integration and deployment, Jenkins is used on an AWS server as we have some experience with it. For the database we are using Microsoft SQL. Ansible is used to set up an appropriate environment on servers to deploy our project on.

For hardware, we are using our own computers, with monitors provided by Síminn. We are also provided with AWS servers to use.

We also set up swagger for API documentation and ease of access to API calls when developing both UI's. Swagger as seen on [api.simamotid.com/swagger](https://api.simamotid.com/swagger).



The image is a screenshot of the Swagger UI for the API at <https://api.simamotid.com/swagger/v1/swagger.json>. The interface is titled "My API" and shows a list of API endpoints under the "Admin" section. The endpoints are:

- GET /api/v1/Admin/sportTypes: Gets all sportTypes
- GET /api/v1/Admin/sportTypes/{sportTypeID}/eventTypes: Finds eventTypes for associated SportType
- GET /api/v1/Admin/event/{eventID}: Announcements for given eventID BOTH HIDDEN AND VISIBLE (DISPLAY)
- GET /api/v1/Admin/announcement/{announcementID}/event/{eventID}: A single Announcement for given eventID BOTH HIDDEN AND VISIBLE (DISPLAY)
- GET /api/v1/Admin/genders: List of genders
- GET /api/v1/Admin/stadiums: Finds all stadiums and Fields associated with them (WILL NOT DISPLAY HIDDEN STADIUMS/FIELDS)
- GET /api/v1/Admin/algorithms: Finds all stadiums and Fields associated with them (WILL NOT DISPLAY HIDDEN STADIUMS/FIELDS)
- POST /api/v1/Admin/wizard: Takes in data from UI Wizard and modulates it to create needed data.

The "Parameters" section shows a single parameter named "model" of type "body". The "Data Type" section shows a JSON schema for the "model" parameter:

```
{
  "eventName": "string",
  "sportTypeID": 0,
  "eventTypeID": 0,
  "startDate": "2017-12-14T19:36:11.991Z",
  "endDate": "2017-12-14T19:36:11.991Z",
  "eventTag": "string",
  "eventDescription": "string",
  "minimumTeamsInGroup": 0,
  "maximumTeamsInGroup": 0,
  "minimumTeamsInSquad": 0,
  "maximumTeamsInSquad": 0
}
```

The "Response Messages" section shows a single response with status code 200 and reason "Success".

---

## 4.2. Build pipeline and Deployment

### 4.2.1. Jenkins

Team members use a Jenkins server hosted on AWS as a build pipeline tool. Six pipelines are currently in use. Two for the general UI itself, one for the master branch, another for all other branches. A couple more for the Admin UI, same as above. Also two for the API, with a similar setup as the aforementioned UI setups.

The API (master) build pipeline, building the code and running tests. If both succeed the project is deployed on deployment server.

The UI (master) build pipeline for both user interfaces, will build the project, and run some unit tests. If successful, it is deployed on deployment server.

The Build pipeline for branches other than master will build the project and run unit tests. If the build pipeline succeeds, a pull request can be created. That request must then be reviewed by another user, and if everything works correctly it is accepted to master branch.

### 4.2.2. AWS

Everything hosted online runs on Amazon Web Services servers, running Ubuntu 16.04. This ensures stability and security, and suites our needs as team members were quite familiar with this environment and running Jenkins on it. These servers were provided by Síminn.

### 4.2.3. Ansible, deployment and hosting

For continuous deployment and hosting environment setup, we ended up using ansible. Ansible is a software to help automate software provisioning, deployment and management. With the help of a Síminn employee, we have created suitable scripts for each project. One to quickly set up the necessary environment to build and run our applications, like installing Angular and .NET. This is only run once against our hosting servers. Another to actually build and run our applications on appropriate ports, and yet another to set up nginx, which is the web server all our applications use, and set up necessary variables for it to work with our applications. The latter two are run every time a push goes on the Jenkins master pipeline, but only of the tests give it thumbs up.

Currently deployed on server with domains:














- [simamotid.com](http://simamotid.com) - general UI to view events

- 
- `admin.simamotid.com` - UI for the admin to create events and manipulate data. Can only be accessed by a user with an account.
  - `api.simamotid.com` is the base url for the api. Documentation of the API can be found on `api.simamotid.com/swagger`

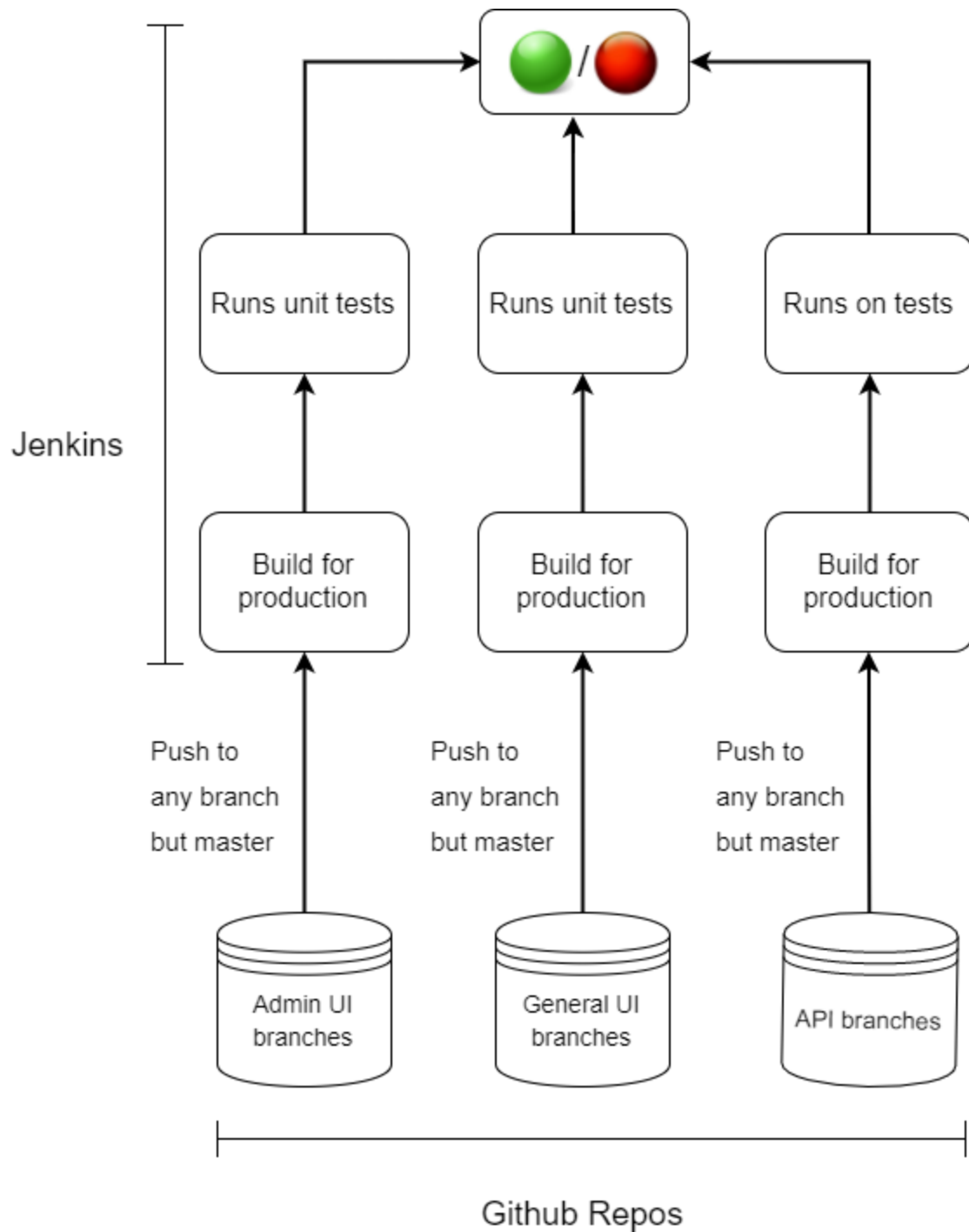
Security is provided via https, which both sites run, and all http requests are re-routed to https automatically. We used Certbot for this, as it integrates seamlessly with nginx and Ubuntu. Auth0 is used for user validation in the admin UI.

#### 4.2.4 Overview of Build pipeline

Our jenkins server uses six pipelines to maintain the entire project, mentioned above. How each pipeline works in detail is explained on the following pages. The image below shows an overview of each pipeline running for our project

Active Items						
		All				
S	W	Name ↓	Last Success	Last Failure	Last Duration	
		<a href="#">Admin-branches</a>	20 hr - <a href="#">#148</a>	23 hr - <a href="#">#146</a>	1 min 46 sec	
		<a href="#">Admin-Deploy</a>	19 hr - <a href="#">#17</a>	2 days 19 hr - <a href="#">#11</a>	2 min 54 sec	
		<a href="#">API-branches</a>	25 min - <a href="#">#144</a>	1 day 0 hr - <a href="#">#136</a>	1 min 8 sec	
		<a href="#">API-Deploy</a>	22 min - <a href="#">#21</a>	20 hr - <a href="#">#19</a>	1 min 35 sec	
		<a href="#">UI-branches</a>	19 hr - <a href="#">#61</a>	14 days - <a href="#">#49</a>	33 sec	
		<a href="#">UI-Deploy</a>	1 day 22 hr - <a href="#">#6</a>	5 days 22 hr - <a href="#">#1</a>	1 min 20 sec	

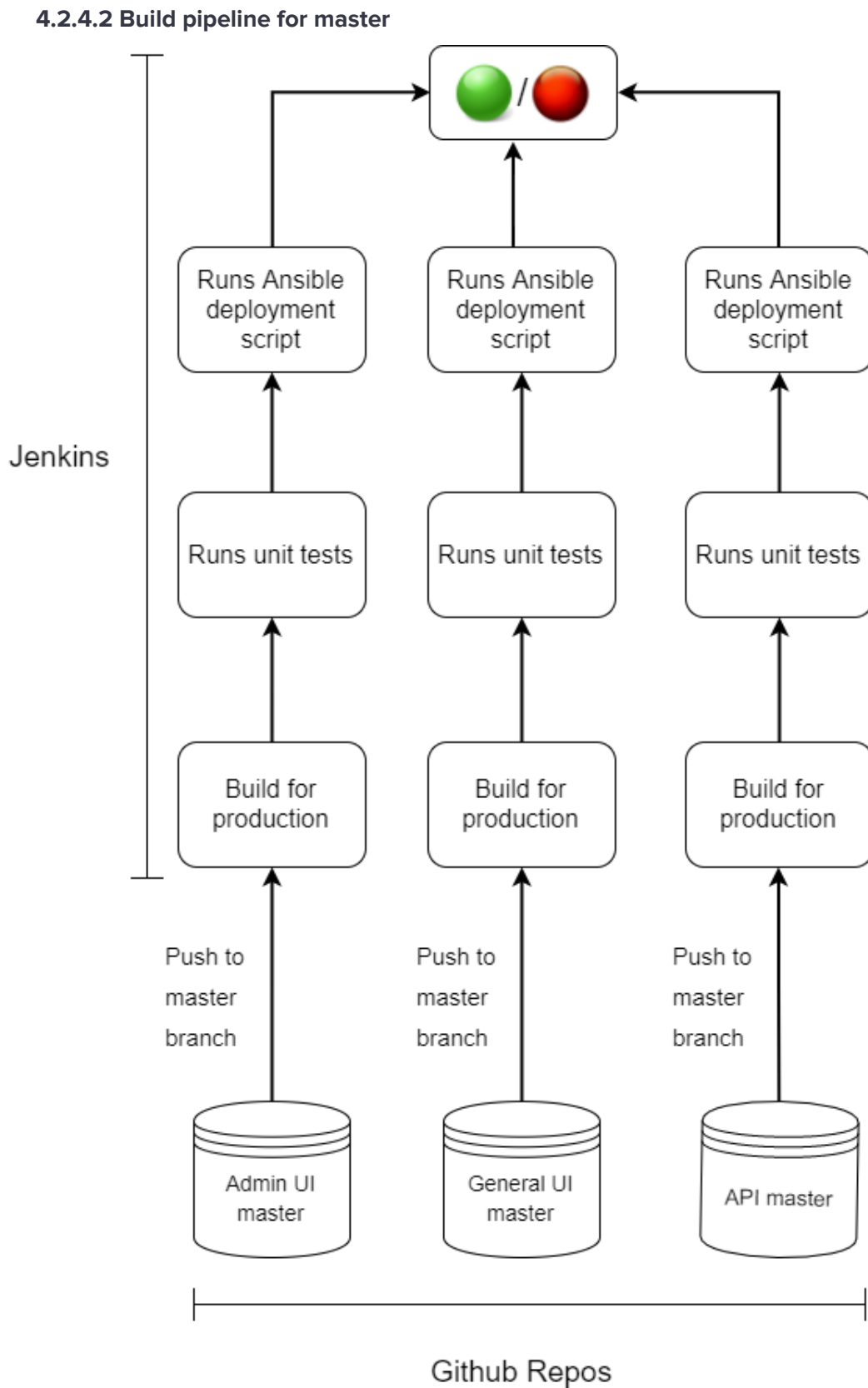
#### 4.2.4.1 Build pipeline for branches



---

The chart above shows how the build pipeline is set up. The branch pipelines are triggered on push to any github branch excluding master. The code is then pulled from github and a build for production is started. This is done to check if the project is production ready. Unit tests are then run for the project and code coverage is generated. Code coverage is not generated for the API, as there is currently no effort from the .NET core development team to do so cross-platform, more specifically for linux.

If any of the steps above fail Jenkins aborts the remaining steps and will display a red ball next to the pipeline that failed. If all the steps complete successfully Jenkins will reward the user with a very satisfying green ball next to the pipeline.



The chart above shows how the pipelines for master are set up. The pipelines are triggered when a change is made to the master branch of each project. Changes are always the result of a pull request from another branch to master. Once the pipeline starts work it pulls all changes from Github for the project and starts building it for production. If the build succeeds the pipeline continues and runs unit tests for the project. Finally the pipeline runs a deployment script. This deployment script is within an ansible project, and a git repository is located on the jenkins server. The script uses ssh to access the hosting server, pulls the current master head, removes the old instance of the project, like the dist directory for angular and publish folder for .NET core, builds a new version from the one pulled from github and if necessary establishes connections to nginx which serves all our projects.

### 4.3. Testing

#### 4.3.1. Unit testing

We strive to have unit tests for most code, which will be run against our code before being submitted to a build pipeline. The unit tests will be run against submitted code in the pipeline as well. Our aim is to have 80% code coverage or higher.

##### 4.3.1.1 General UI tests

Unit tests conducted by the general user interface are few but cover up a lot of the code, since there isn't much logic to be had excluding the html and css. The general user interface has a total of 22 tests. Those tests cover a total of 74.27% of statements as indicated by our code coverage report visible on Jenkins. Branches (else statements etc.) have about 20% less coverage but this is mainly due to the fact that the Karma code coverage generator accounts for else statements for if statements that have no else. Which the UI has a bunch of.

[Back to UI-Deploy](#) [index](#)

### All files

74.27% Statements 583/785 52.11% Branches 136/261 59.57% Functions 112/188 73.37% Lines 529/721

##### 4.3.1.2 Admin tests

Unit tests conducted by the admin user interface are more robust than the general user interface. The admin user interface has 60 written unit tests. Those

tests however only end up covering 59.94% of the statements written for the Admin user interface and around 45.72% of the branches. This user interface contains a lot of validation logic that has to be tested and hasn't fully been explored.

[Back to Admin-Deploy](#) [index](#)

## All files

**59.94%** Statements 1393/2324   **45.72%** Branches 342/748   **41.31%** Functions 221/535   **59.45%** Lines 1296/2180

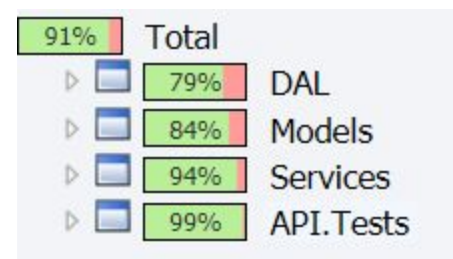
### 4.3.1.3 API tests

Api testing was done using the tool Xunit. The tests were split into two different types of tests. One type tests the return values from the controllers in the API, and the other type tests functionality called from the service. Both classes of tests needed to test functionality to be of any use, however the main focus was different. The philosophy that was used was to avoid isolated and therefore not useful tests, but for each test to test real functionality, even though that meant testing more than one method at a time.

The controller tests make sure that the correct status code and value is returned given the input data. It tests if methods return correct status codes for data not found, illegal or incorrect data, failure on the server side and of course test for a successful return.

The service layer tests try to go deeper into the logic of the code, testing to see if data is created in the correct manner, if associations between entities are correct after changes to database, to make sure that changes will result in the changed data being returned when called by other methods and so on.

The combination of these tests result in a code that is comprehensively tested with 133 individual tests (and counting), many of which test complicated behaviour patterns. Code coverage for the API modules is between 79% and 99%.



## 5. Progress Report

### 5.1. Sprint 0

Sprint zero's purpose was to analyze the project requirements and decide which of those was most important to implement. During this sprint, the team members got an extensive idea of what the project should be and further split those ideas down into requirements.

#### 5.1.1. Working draft

We estimated around 16 working hours per week per team member, and with this sprint being around one and a half week that put estimated time at 24 hours. In reality we spent around 30 hours, which we felt was acceptable and necessary for what had to be done in this sprint. All in all, this puts us right around schedule, with non-functional requirements being the only thing that we did not manage to fully realise this sprint.

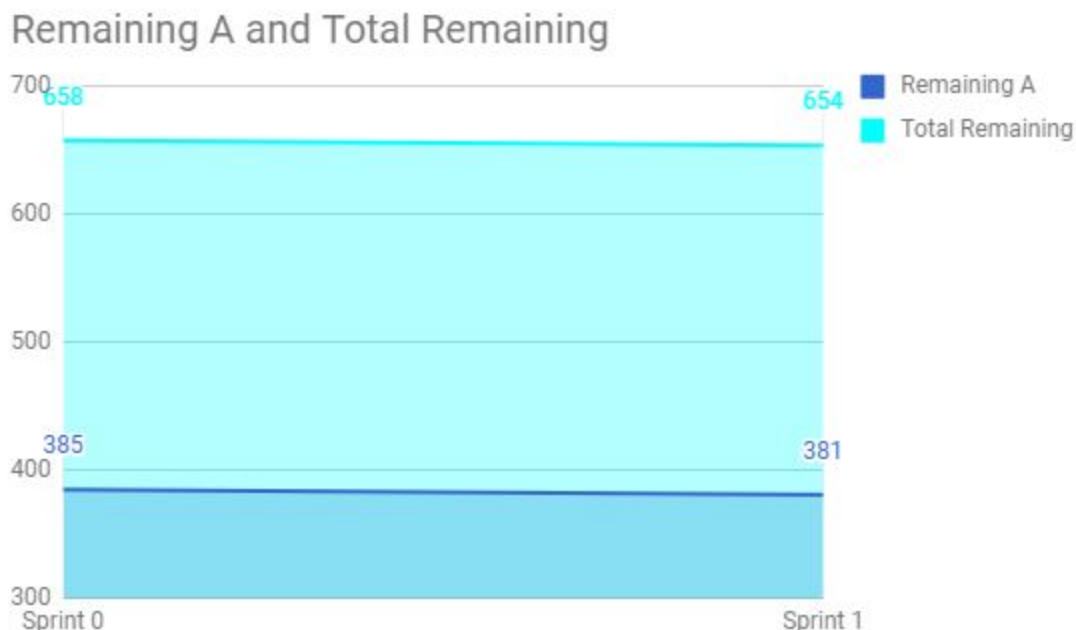
#### 5.1.2. Retrospective

What went well	What could have been improved	Actions to take
Communication between team members went smoothly, and the atmosphere is delightful	Regular work hours could have been established in the beginning to ensure work was done	Keep communications light and keep discussing all ideas came up
No idea went ignored and undiscussed	Time spent on assignment could have been used more efficiently during work hours	Enforce regular work hours and ensure that those hours are well spent.
If an idea is discarded there are no hard feelings between members		

## 5.2. Sprint 1

Sprint one's purpose was to Set up the working Environment, understand the requirements and required specifications, speak to experts about the issues we might face, analyze database and more.

### 5.2.1. Burndown Chart



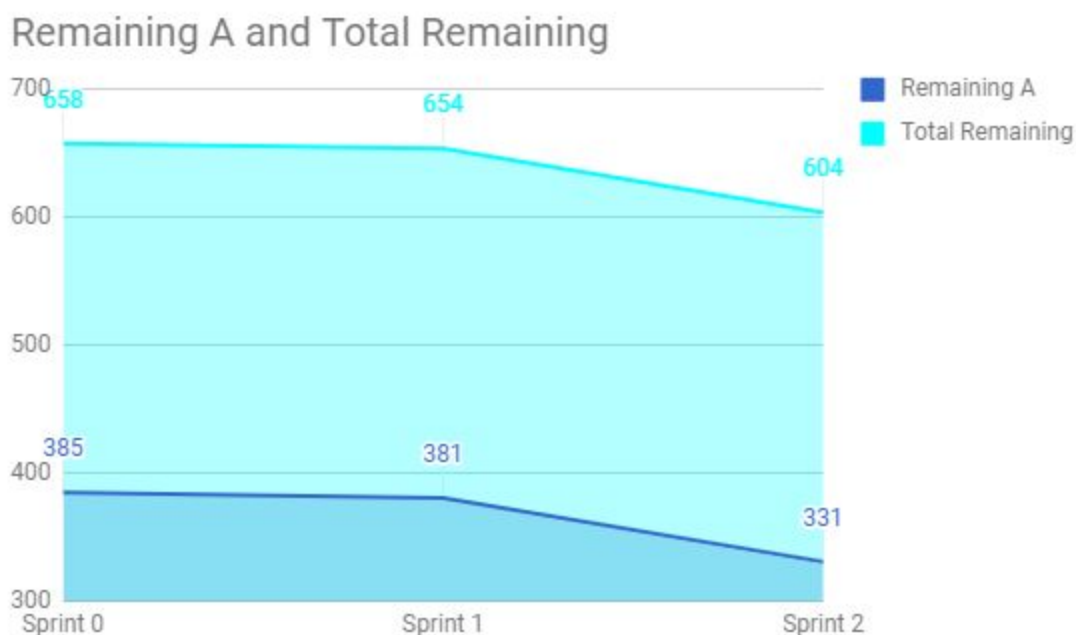
### 5.2.2. Retrospective

What went well	What could have been improved	Actions to take
As mentioned in the last sprints retrospective communication has been excellent	Regular work hours were not enforced enough,	As in the previous sprint work hours have to be forced unless specifically planned work hours are changed.
Time management while working went smoothly		

### 5.3. Sprint 2

Sprint Two's purpose was to create a working Entity Framework Schema, Connection to Database, Create important Entity Controllers and more.

#### 5.3.1. Burndown Chart



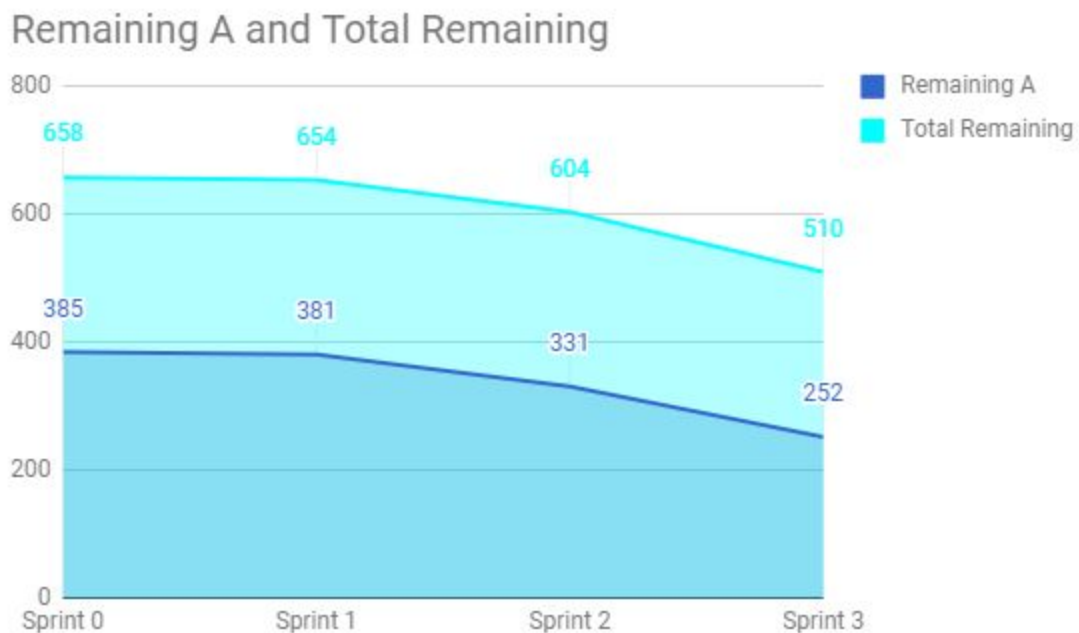
#### 5.3.2. Retrospective

What went well	What could have been improved	Actions to take
Communication went well between team members	Time management could have been better, went worse than last sprint	Reestablish better time management
Discussions about implementation details were really effective	Regular work hours still not enforced enough	Be more strict about work hours

## 5.4. Sprint 3

On the last day of the sprint, we realized that we needed to greatly overhaul the requirements list, seeing as it did not fulfill all the requirements we found necessary for the MVP. A greater problem was that the existing requirements were way too general to know exactly what was expected from each requirement, so they were broken down into smaller subtasks.

### 5.4.1. Burndown Chart



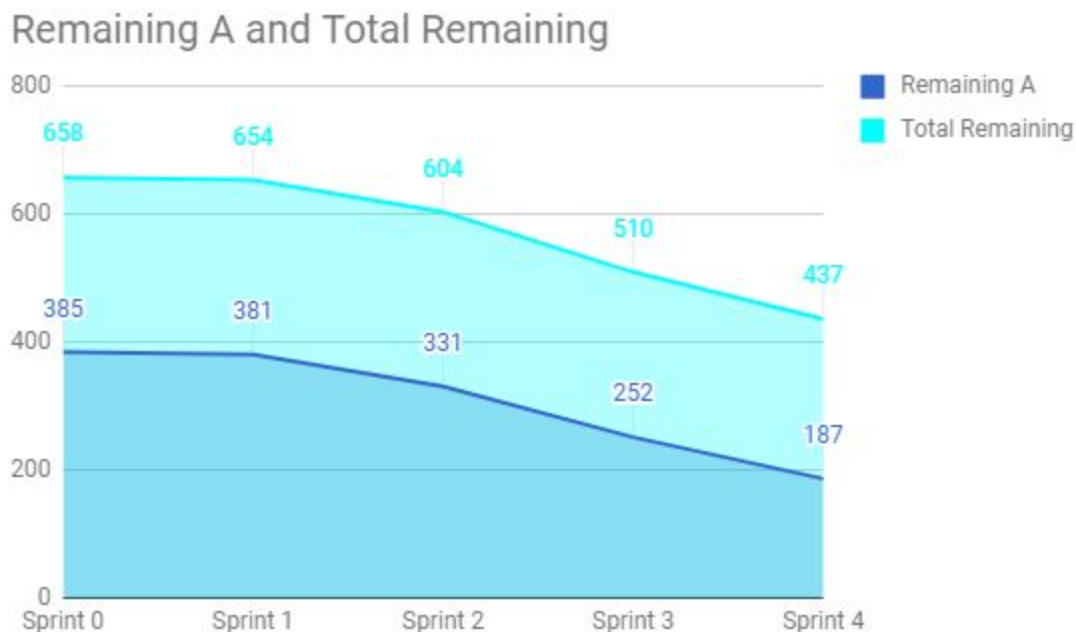
### 5.4.2. Retrospective

What went well	What could have been improved	Actions to take
Team effort, and realizing sooner than later that the requirements list needed an overhaul.	Point estimation in the requirements list.	Analyze requirements list and add new requirements as we see fit.

## 5.5. Sprint 4

After making changes to the requirements list at the start of this sprint, we realized that we had more work cut out for us than we realized. Here we tried as we could to finish as many points as possible before exams started.

### 5.5.1. Burndown Chart



### 5.5.2. Retrospective

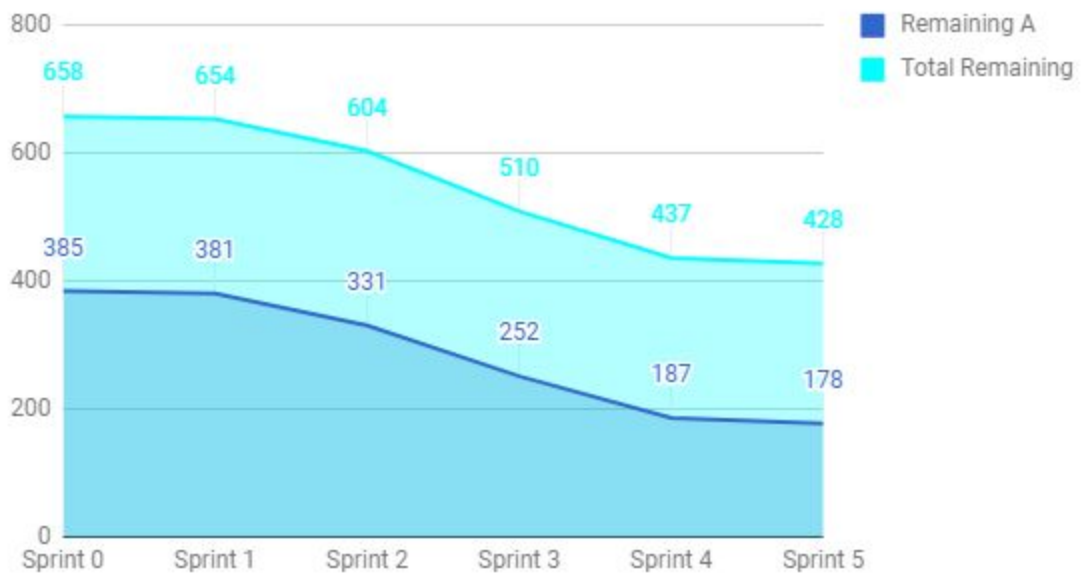
What went well	What could have been improved	Actions to take
Finishing the points left over from sprint 3.	We should have set up authentication sooner.	Better realize what other requirements need to be finished sooner than later.

## 5.6. Sprint 5

We estimated that very little work could be done as this sprint took place right around exams, so only 24 points were available to complete during this time, and we only assigned 9 points for us to complete.

### 5.6.1. Burndown Chart

Remaining A and Total Remaining



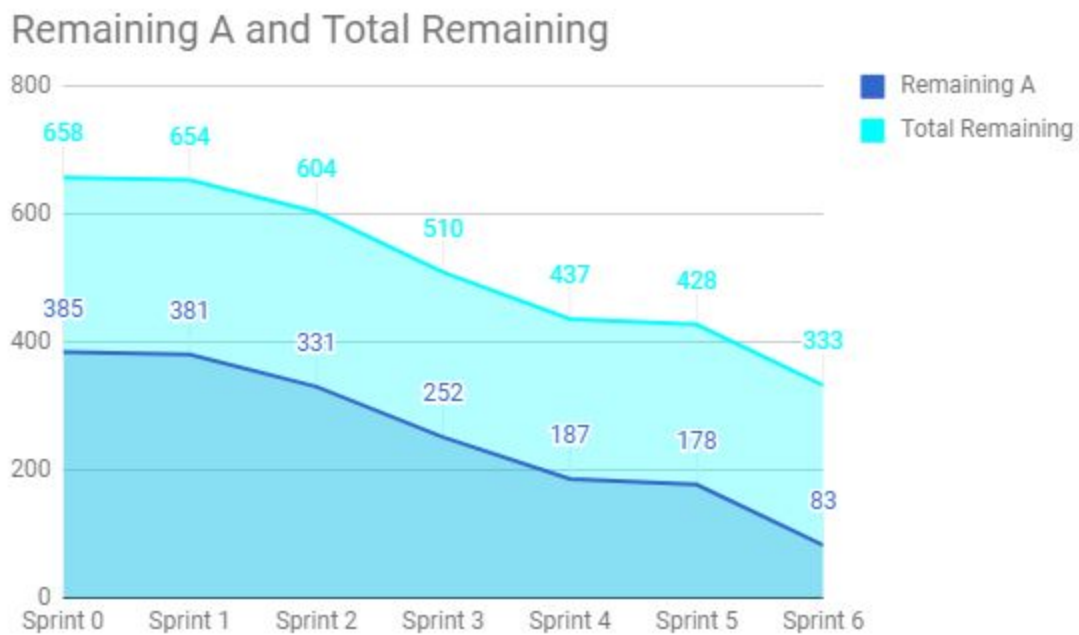
### 5.6.2. Retrospective

No retrospective necessary for the amount of work accomplished during this sprint.

## 5.7. Sprint 6

This is when we started to break the sprints down into a week each instead of the 2 week sprints used so far, as much greater time could be allocated to the final project. Near the end of the sprint, a scrum board was set up at Síminn, for which we used to monitor our progress better.

### 5.7.1. Burndown Chart



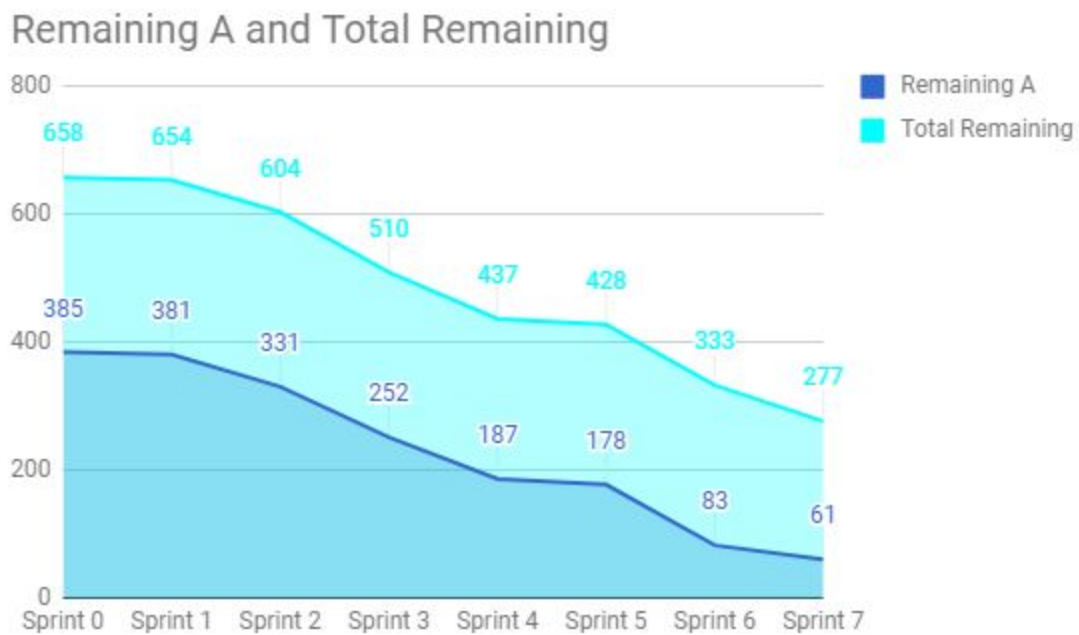
### 5.7.2. Retrospective

What went well	What could have been improved	Actions to take
Using the available time well to finish tasks.	Estimating the time each requirement takes.	Improve the requirements list.
Using the scrum board to improve workflow.	Starting to use a scrum board sooner than we did.	Start implementing the use of a physical scrum board in future projects.

## 5.8 Sprint 7

With much more time allocated to working requirements, a great amount was done. Although an average amount of requirements were finished in this sprint, very important work has been done that will make future story points easier to finish. Continuous Deployment was also implemented in this sprint.

### 5.8.1. Burndown Chart



### 5.8.2. Retrospective

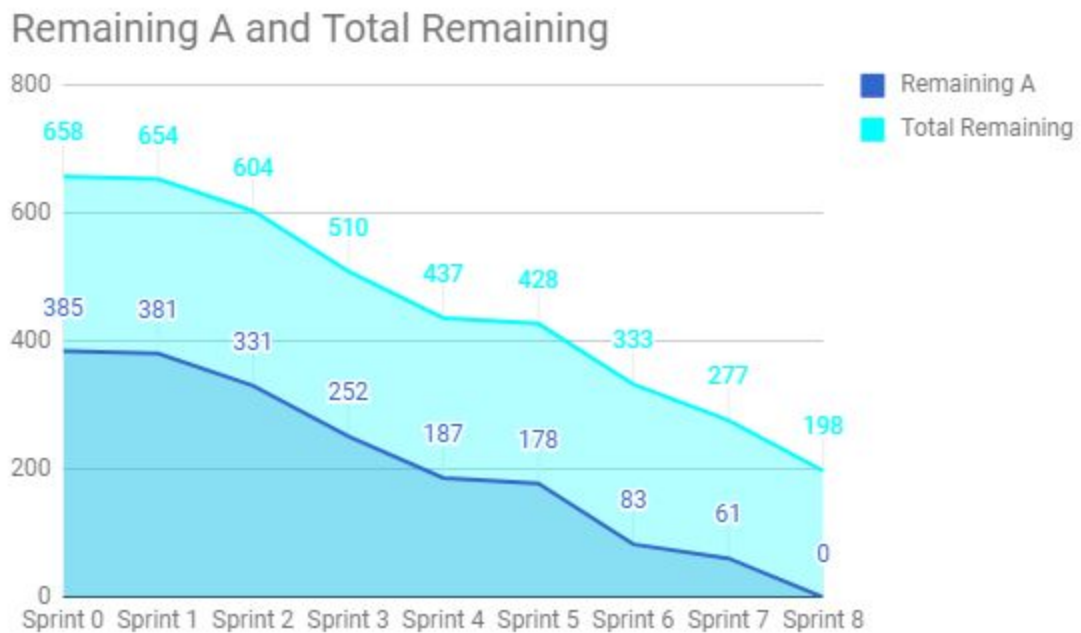
What went well	What could have been improved	Actions to take
A great amount of important work was done	Better documenting	Reserve more time to document changes
Scrum assisted in guiding the path of the project		
Extremely good cooperation		
Difficult tasks finished quickly		
Good feedback and assistance from Síminn Employees		

## 5.9 Sprint 8

An extremely productive sprint. All A-Requirements were finished as well as a few B-requirements. The work from the previous sprint was very useful because it helped speeding up the completion of the last remaining A-Requirements.

Code freeze began on 13. December, 4 days before the end of the sprint.

### 5.9.1. Burndown Chart



### 5.9.2. Retrospective

What went well	What could have been improved	Actions to take
All A-requirements were finished	In almost all respects, the sprint could not have been improved	Keep working well as a team
Code freeze on time		
Team created a working product		

## 5.10 Overview of Sprints

The total amount of time spent on the assignment is around 996 hours. In the last three sprints (6,7,8) exactly half of the total finished requirements were finished (230 of 460).

Compared to the story points assigned to each user story the amount of time spent on each story does not reflect the work done at all. Most stories took twice the amount of estimated time to complete.

### 5.10.1. Finished requirements over time

	Finished A	Finished B	Finished C	Total Finished	Remaining A	Remaining B	Remaining C	Total Remaining
Sprint 0	0	0	0	0	385	199	74	658
Sprint 1	4	0	0	4	381	199	74	654
Sprint 2	50	0	0	50	331	199	74	604
Sprint 3	79	10	5	94	252	189	69	510
Sprint 4	65	8	0	73	187	181	69	437
Sprint 5	9	0	0	9	178	181	69	428
Sprint 6	95	0	0	95	83	181	69	333
Sprint 7	22	34	0	56	61	147	69	277
Sprint 8	61	18	0	79	0	129	69	198

### 5.10.2 Time spent on assignment

As of 14.12.2017 each team member has worked the total of hours mentioned below. Totalling to 994 worked hours on the final project.

Team member	Hours spent
Arnar Freyr Sævarsson	349
Barði Freyr Þorsteinsson	317
Valdimar Jónsson	328
<i>Total hours</i>	<i>994</i>

## 6. Requirements

### 6.1. Explanation

Label	Definition
S.n.	The serial number of the requirement.
P	Priority of the requirement. Can be A, B or C depending on importance.
UI	Points specific to the UI of the project.
API	Points specific to the API of the project.
S	In which Sprint the story was completed.
Bold items	An epic, or a story with many sub-stories below.

### 6.2. Functional Requirements

\*Green requirements indicate that it has been finished

S.n	Category	I want to	So that	P	UI	API	S
1	<b>Events</b>	<b>View information about sporting events</b>	<b>I can keep up with events I have interest in and information about those events</b>	<b>A</b>			<b>2</b>
2	Events	View the teams competing in that event	I can know what I can expect of upcoming matches	A	2	2	3
3	Events	View upcoming matches of a specific event	I can view when my favorite teams are competing	A	1	2	3
4	Events	View the status of those events	I can keep track of each teams status in the event	A	1	1	2
5	Events	View the event sport type and name	I know what the event is about	A	1	1	3
6	Events	View the start date and end date of that event	I know that events date range	A	1	1	3
7	Events	View ongoing matches	I can view ongoing matches	A	1	1	3
8	Events	View finished matches	I can view finished matches	A	1	1	3
9	<b>Matches</b>	<b>View information about a single match</b>	<b>I can be sure of what I need to know about a given match</b>	<b>A</b>			<b>2</b>
10	Matches	See the location of that match	I can know where to show up if I want to watch the match	A	1	1	3
11	Matches	See the teams competing in that match	I can know who the competitors are	A	2	2	3

12	Matches	See the result of that match (if score is being tracked)	I can know who won and who lost	A	1	1	3
13	Matches	See when that match is scheduled (starts, ends)	I know when to start watching/when to arrive	A	1	1	4
14	Matches	View the status of different Squads in a tournament	I can view Squads in Sections that interest me	A	2	4	4
15	Matches	View groups for an event that interests me	I can know what matches will be played there	A	2	4	3
16	Matches	View Sections for the events that interest me	I can know what matches will be played there	A	2	4	3
17	<b>Events</b>	<b>Create a new event</b>	<b>My event will be in the system for users and for ease of event maintenance</b>	<b>A</b>			
18	Events	Specify what type of sport the event is related to	Users know what kind of event it is	A	3	3	4
19	Events	Maintain my existing events	My event will be correct in the system for users and for ease of event maintenance	A	2	4	4
20	Events	Change my existing events - Edit any information that may be wrong/misleading	My event will be in the system for users and for ease of event maintenance	A	3	20	6
21	Events	Cancel my event(s).	Users can know whether an event is still on	A	1	6	8
22	Events	Change teams participating in event	Only teams that are to participate in the event are visible and assignable to matches	A	6	8	6
23	Events	Choose an algorithm for each event section	Matches and teams are ordered correctly(Only elimination algorithm in MVP)	A	1	16	7
24	Events	Change event sections	Correct sections are present	A	4	8	6
25	Events	Change Groups within an event	Correct groups are present	A	3	8	6
26	Events	Wizard view model for administrator	A new event base structure can quickly be set up	A	24	12	4
27	Events	Wizard-API connection	The wizard can confirm if data is correct before it is submitted to database	A	1	3	4
28	Events	Edit event section timing	The correct time can be displayed	A	3	2	4
29	Events	Change stadiums/fields in an event	Only fields that are to be used in the event are visible and assignable to matches	A	6	2	6
30	<b>Matches</b>	<b>Edit any match in my event/s</b>	<b>Errors can be corrected</b>	<b>A</b>			
31	Matches	Edit the score of a match	The score can be corrected if the score was wrongly input	A	1	4	7
32	Matches	Automatically have squads set up for given section	I know what teams should play together	A	2	32	8
33	Matches	Edit automatically generated squads	I can make changes that better suit the needs of the event	A	2	8	8

34	Matches	Remove entire squads	I can fix any mistakes or errors that were inserted	A	2	8	8
35	Matches	Edit Groups of teams	I can fix any mistakes or errors that were inserted	A	2	8	6
36	Matches	Remove Groups of teams	I can fix any mistakes or errors that were inserted	A	2	8	6
37	Matches	Create another admin	More admins can control the system	A	1	2	5
38	Administration	Create a field	Event organizers can organize matches on correct fields	A	1	6	6
39	Events	Add event Announcement	Announcements will be displayed on UI front page	B	8	4	8
40	Events	Edit/Delete event Announcements	Announcements will be displayed/removed on UI front page	B	2	4	8
41	Matches	Edit the location of a match	The match location can be corrected if the location is changed	B	1	2	
42	Matches	Edit the start time of a match and match length	The match start time can be changed/corrected	B	1	4	
43	Matches	Cancel matches	Users can know whether a match is still on	B	1	6	
44	Matches	Revert scores to matches in progress	Mistakes can be corrected as soon as they appear	B	1	8	
45	<b>Matches</b>	<b>Filter matches</b>	<b>I can find matches that matter to me</b>	<b>B</b>			
46	Matches	See other match statistics (shot attempts, fouls etc.)	I can know what happened during the match.	B	2	3	
47	Matches	Filter by Day	I can find matches that matter to me	B	1	1	4
48	Matches	Filter by Field(Venue)	I can find matches that matter to me	B	1	1	4
49	Matches	Filter by Team	I can find matches that matter to me	B	1	1	4
50	Matches	Filter by Event	I can find matches that matter to me	B	1	1	4
51	Account	Register an account	I can access features that are unavailable to unregistered users	B	1	10	
52	Account	Log-in	I can better monitor events and matches I am interested in	B	1	4	
53	Account	Subscribe to an Event (if logged in)	I can keep track of specific events I have more interest in than others	B	1	1	
54	Account	Delete account (if logged in, soft delete)	I can remove unwanted information	B	1	2	
55	Matches	View my assigned matches	I know when I'm assigned/what I'm assigned to	B	4	2	
56	Matches	Be assigned to a specific match	I can make sure information is correct	B	3	2	
57	Matches	Be able to edit my assigned matches, after they are closed	I can fix any mistakes or errors that were inserted	B	4	3	

58	Events	Associate users of any usertype (excluding general users) to the event	The event has all required personnel are connected to the event and can add/insert relevant information about its matches	B	1	16	
59	Matches	Add scores to matches as it happens (with timestamp)	Users can view a feed of scores in the game	B	8	8	
60	Matches	Add a message about something happening in the match as it happens (with a timestamp)	Users can view a feed of events in the game	B	8	8	
61	Administration	Delete any user(Soft delete)	The system isn't cluttered with unnecessary/invalid users	B	2	6	
62	Administration	Create a referee user	Referees can keep track of their matches and the stats of those matches	B	1	1	
63	Administration	Create a staff user	More people can update the status of games and the tournament to ease the responsibility of the event organizer	B	1	1	
64	Development	Create a controller for Fields	I can work on other user stories	C	2	3	3
65	Matches	View the weather forecast at match location	I can know what I need to bring for different weather.	C	1	2	
66	<b>Teams</b>	<b>View team information</b>	<b>I can monitor teams I am interested in</b>	<b>C</b>			
67	Teams	View team coach	I can see who is currently coaching the team	C	2	2	
68	Teams	View the individual player info	I can see details about players in the team	C	2	2	
69	Teams	View whole team player roster	I can see who are currently playing for the team	C	2	2	
70	Teams	View upcoming matches the team are to participate in	I can better monitor a team I am interested in	C	2	2	
71	<b>Teams</b>	<b>Manage Team Information</b>	<b>Information about team will include correct players and other information.</b>	<b>C</b>			
72	Teams	Register my team	I can be able to manage my teams information and apply for tournaments	C	2	3	
73	Teams	Manage player information	Information about my team members is up to date	C	2	3	
74	Teams	Manage coach information	Information about my teams coaches is up to date	C	2	3	
75	Teams	Add and remove players from the team	I can change the team player structure as I see fit	C	2	4	
76	Teams	Add and remove coaches from the team	I can change the team coach as I see fit	C	2	4	
77	Teams	Change team rating	I can better manage how good I think my team is at a given time	C	2	3	
78	Teams	Register Team for a tournament	Playing teams are visible in the software for users to see	C	2	4	

79	Events	Add a set of different rules to my event to specify the result calculations	I can have full flexibility of how future matches for my event are decided	C	3	7	
80	Administration	Create a team organizer	A team organizer can maintain their team info and events they participate in	C	1	1	

Number of A requirements (including subtasks): **54 (were 27)**

Number of B requirements (including subtasks): **30 (were 18)**

Number of C requirements (including subtasks): **17 (were 16)**

- In the end of the third sprint we realised that the requirements assigned were not sufficient enough for our **MVP**. We ended up breaking down multiple A requirements and re-prioritising many of the requirements as they were not necessarily needed for the system to work as a whole, along with adding a few as we saw fit.
- The requirements above include development requirements that are not mentioned in the requirement list above

## 6.2 Non-functional requirements

No.	Requirement	Priority
1	Keep latency of POST request at a low 200ms maximum	A
2	>80% of code is unit tested	A
3	The system is scalable so that adding new types of sporting events, score calculations is simple and efficient	A

## 7. Diary

Day	Arnar	Barði	Valdi
01.09.2017	Planning and Preparation	Planning and Preparation	Planning and Preparation
05.09.2017	Planning and Preparation	Planning and Preparation	Planning and Preparation
08.09.2017	Planning and Preparation	Planning and Preparation	Planning and Preparation
11.09.2017	Planning and Preparation	Planning and Preparation	Planning and Preparation
14.09.2017	UI Work	API created, basic functionality implemented	Root UI Project created
18.09.2017	UI Work	Work on database as well as API connection to it	Webhook, Karma testing in UI
21.09.2017	UI Components created, Banner and Navbar	API Controllers	Work on build script for API
25.09.2017	UI Work		UI Work, Testing Docker platform
28.09.2017	UI Components and testing		Server work, bash file created
02.10.2017	UI Views and routing	Server Connection established (API), Basic Admin UI implemented	Dev script, updated readme (API)
05.10.2017	UI Components	API Controllers	Server work
09.10.2017	UI Components	API Controllers	Server work
12.10.2017	Unit testing and Component work	API Controllers	Swagger added to API, Cross-Origin Requests. Code coverage testing for UI
16.10.2017	UI connection to API	API Controllers, Middleware created	UI connection to API
18.10.2017	UI connection to API	Fixing and implementing API Calls	UI connection to API, Deployment
19.10.2017	Evaluation of specification	Evaluation of specification	Evaluation of specification

23.10.2017	Work on Admin UI (Wizard) and finishing touches to User UI	Work on API calls	Fix UI Elements
26.10.2017	Work on Admin UI (Wizard)	Work on Admin API functionality	Fix UI Elements and announcement connection
30.10.2017	Work on Admin UI Authentication	Work on API unit tests	Work on Authentication (Research)
02.11.2017	Work on Admin UI Authentication	Work on API Authentication	Work on API Authentication
06.11.2017	Connection of Admin API to wizard	Work on API Authentication	Work on Admin UI Announcement edit
27.11.2017	Work on Admin UI (Edit/Delete) Teams	Work on Admin API functionality	Work on Admin API calls
28.11.2017	Work on Admin UI (Edit/Delete) Teams	Work on API unit tests	Work on Admin API Calls.
29.11.2017	Work on Admin UI (Edit/Delete) Fields	Work on Admin API functionality	Work on Admin API functions Information
30.11.2017	Work on Admin UI (Edit/Delete) Fields	Work on Admin API functionality	Work on Admin UI (Edit/Delete) Information
01.12.2017	Work on Admin UI (Edit/Delete) Sections	Work on API unit tests	Work on Admin UI (Edit/Delete) Information
02.12.2017	Work on Admin UI (Edit/Delete) Sections	Work on Admin API functionality	Work on Admin UI (Edit/Delete) Groups
03.12.2017	Unit testing of Admin UI and mocking	Work on API unit tests	Work on Admin UI (Edit/Delete) Groups
04.12.2017	Work on Report & user testing	Work on Report & user testing	Work on Report & user testing
05.12.2017	Work on Admin UI Sections	Unit tests	Worked on adding announcements to the UI.
06.12.2017	Work on Admin UI Sections	API functionality	Fixed connection with API in UI. Ansible setup and testing.

07.12.2017	Work on Admin UI Squads	API Algorithms	Bug fixes in UI. More ansible testing.
08.12.2017	Work on Admin UI Squads	API Algorithms	Added a lot of validation in Admin UI.
09.12.2017	Work on Admin UI Matches	API Algorithms	Admin dropdown fix. Fixed ansible.
10.12.2017	Work on Admin UI Matches	Code comments, API functionality, bug fixes	Algorithm connection to api from Admin view.
11.12.2017	Bug fixes	Unit testing, bug fixes	Fixed key permissions in ansible. Bug fixing in bot UI and Admin UI.
12.12.2017	Report work	API methods functionality, Report work	Report work
13.12.2017	Code cleanup	Code cleanup	Code cleanup
14.12.2017	Report work	Report work	Report work