Mälardalen University
School of Innovation, Design and Engineering
Västerås, Sweden

Software Engineering 2 : Project Teamwork - 7.5 hp - DVA313

BLACK RIVER RUN TIME KEEPING

Project Plan

Bastien Delbouys
bds18002@student.mdh.se

Zacharias Claesson
zcn16001@student.mdh.se

Sebastian Oveland
sod16003@student.mdh.se

Cécile Cayèré
cce18001@student.mdh.se

Mohammed Abuayyash
mah18005@student.mdh.se

Rikard Gestlöf
rgf16001@student.mdh.se

Johannes Sörman
jsn16009@student.mdh.se

November 27, 2018

Contents

1 Introduction

Every year in Västerås there is a running competition called Black River Run. The runners are running with a SI-Unit (time tracking unit) that sends its data to a server which is later fetched by a webpage and made available for all that visits the webpage.

The current website is outdated and in need of an update. Both in terms of internal and external qualities. The internal qualities are for instance the structure of the current database in use and the front-end code and back-end code of the webpage. The external qualities are in this instance, the interface and security of the webpage. Another external quality is the user-friendliness and responsive aspects of the front-end of the webpage.
The group will develop this new updated website. A new database that stores runners and races information will also be developed. The SI-Unit and the third-party webserver (OLresult) will not be changed in this project.

This report aims to introduce the different people involved in the project. This report also describes the organization between members of the project group and the processes of communication and work in the team. It enumerates the risks and challenges concerning successful project delivery and provides solutions. It lists and describes the requirements for the project. The report outlines the time plan which includes the deliverables deadlines, the differents activities and planned hours by activities and by members. Finally, it introduces the first view on the development of the project.

2 People Involved

2.1 Project Group

Name	**Role**
Bastien Delbouys	Project manager
Zacharias Claesson	Client contact responsible
Sebastian Oveland	Configuration manager
Cécile Cayèré	Document responsible
Mohammed Abuayyash	
Rikard Gestlöf	
Johannes Sörman	

2.2 Steering Group

Name	**Role**
Jan Carlson	Responsible teacher Examiner
Robbert Jongeling	Course assistant

2.3 Client

- **Name :** Christoffer Holmstedt
- **Organization :** Västerås Running Club
- **Email :** christoffer.holmstedt@gmail.com
- **Phone number :** +46 (0)73 7816126

3 Organization

The group will meet at least once a week on Mondays before the weekly meeting with the steering group. During this meeting the group will discuss the work done, the advancement on the project as well as the plan for the upcoming week. The main tool that the group uses for communicating with each other is Microsoft Teams. Microsoft Teams runs on multiple platforms and has many tools other than communication that the group utilizes.
In detail, Teams allows :

- Messages exchanges.
- File storage and collaborative editing.
- Establishment of a Working Plan.
- Files inspection (many file format are compatible with it).
- Send reminders, notification, mails.

To keep track of people's work, the members has to report their individual work hours. The project manager adds common group hours (hours where the group works together as a unit).
Some files are dedicated to keep track of people's communication. Whenever there has been a group meeting, one member documents the discussed topics. This is because everything that have been agreed upon needs to be written down, so no misunderstanding occurs. It is useful if a member do not remember something that was said or if members are not present at the meeting. To summarize, the documentation is required to define and reform text to implementation.

The choice for storing the implementation components is GitHub. The GitHub repository is structured by folders as followed :

- **Documentation :** It contains the documentation required to be written (e.g. Project Plan Document).
- **GroupMembers :** It contains member names.
- **Presentation :** It contains every presentation done as PDF files.
- **Project :** It contains contains the code of the project.

For document creation, the group will use LaTeX.
As a first step, group members will be assigned part of the document to write. Once the different parts have been written, the document responsible will aim at integrating the different parts into the final LaTeX report.

The presentations will be created using Microsoft PowerPoint on Teams.

4 Current Black River Run Program

This section provides an explanation of the current Black River Run program.

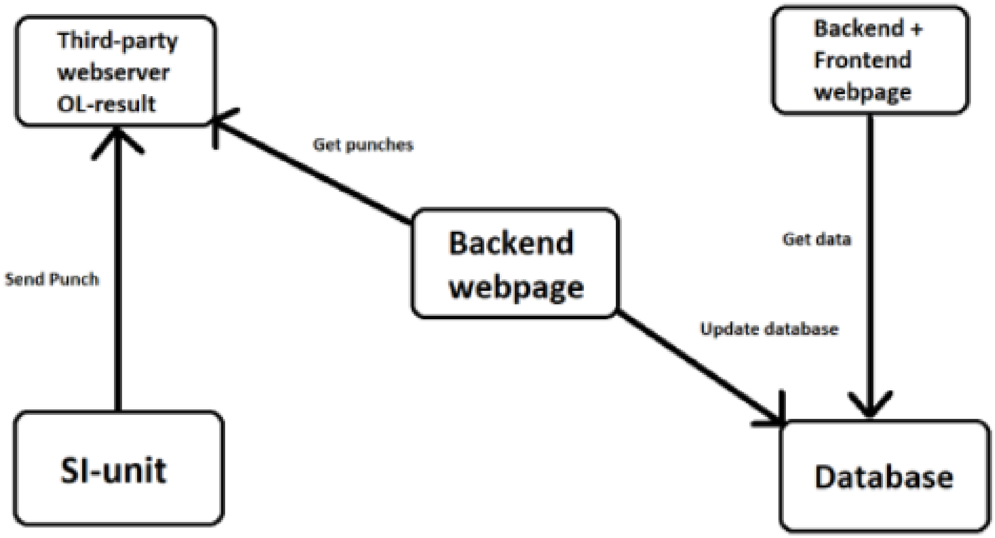

Figure 1: Current Black River Run System.

The website gathers the timestamps, also called punches, for each runner from a third-party server (OLresult) and saves it to the webservers database. Each station on the race route and start/finish station has a unique number that is stored with a timestamp and a SIUnit number.

The system associates every SIUnit with a runner and use these timestamps to calculate the elapsed time.
The client talked about problems that can come up under the race. For example, the API can hold the punches for a long while and send them at a later point. This makes it seem like a person have not come that far yet, but in fact possibly passed that station a long time ago. This means that when the race is active there can be some poorly updated data. When this happens, the administrator must wait for the race to end to get all the timestamps registered in the database. Another problem is that timestamps can be too close to each other, since every runner probably are going to get a lot of punches when they pass a station. This will be solved by the system by removing timestamps for a runner that are too close in time.

Virtual punches are information about the runner that is sent to the third-party webserver. This information includes unitId, lastID, date, time, PunchID, ControlCode, SINumber and SITime. The following comes from a document provided by the client :

- **unitId :** It is the unique identifier of the Radio Online Control.
- **lastId :** It contains the last PunchId that is received so the data is not sent twice.
- **date :** It is the date of the punch according to the Radio Online Control. Its format is *YYYY-MM-DD.*
- **time :** It is the time of the punch according to the Radio Online Control. Its format can be *HH:MM:SS*, *HH:MM* or *HH.*
- **PunchId :** It is incremented by 1 for every punch.
- **ControlCode :** It is the control code that specifies if the punch is a Check, Control, Finish, Start or Clear.
- **SINumber :** It is the SI number.
- **SITime :** It specifies the time the punch was done according to the SI-master.

5 Quality Assurance

There are several risks that exists that can affect the project quality.

One risk is that the group communicates poorly with each other, which could impact the work. The group consists of several nationalities which could contribute to misunderstandings. Also, the majority of the members have never worked with each other which could have the same effect.
To avoid this the group members needs to be very clear on what they mean when they are trying get their points across. The group should also arrange face to face meetings on a weekly basis to discuss anything that is unclear.

A second risk is that the group communicates poorly with the client. This could lead to a product that does not meet the requirements.
To avoid this the group has come to an agreement with the client that he should be informed about the work done on a weekly basis.

A major risk to the product is that the requirements are poorly defined and hence the final product won't live up to the expectations of the client.
To avoid this problem the group should discuss the requirements with the client and the steering group to validate them.

The group members have different skills and varying experience levels which could impact the time-schedule.
This needs to be taken into account when planning the allocation of group members to work on different tasks.

6 System Users

The program will deal with two types of users. They are shortly described in this part but the report will show more about them later.

- **Guest** can view races and runners information.
- **Administrator** can manage races and runners.

7 Requirements

7.1 Functional Requirements

This section provides the functional requirements as user stories.

Identifier	As a...	I want to...	So that I can...
1	Guest / Administrator	See the list of races.	Choose one of them and see the race categories.
2	Guest / Administrator	Choose a race category.	See runners information for this race category.
3	Guest / Administrator	See the list of runners participating in a race.	See race information about them (place, timestamp, name, team, ...).
4	Guest / Administrator	Search for specific runner by entering keyword (name, bib, sse, team ...).	See the list of runners matching to the keyword.
5	Guest / Administrator	Select a runner in a specific race.	See their results over the race and more details about them.
6	Guest / Administrator	Sort runners information in ascending/descending order.	Find a runner easily, depending on their name, place, bib, ...
7	Administrator	Edit race data.	Manage information about a specific race (correct mistakes, change a date, add more details...).
8	Administrator	Edit runner's data.	Correct errors due to the hardware used during the race or change user's information.
9	Administrator	Create a new race.	Add new participant runners to the race.
10	Administrator	Add a new runners to a race.	Make the connection with the hardware used and receive information from it during the race.

7.2 Non-Functional Requirements

One non-functional constraint on the system is that the database can only have up to 200 active connections simultaneously.
Another constraint is that our system is relying on the OLresults API for the runner information.

8 Use Cases

This section allows to formulate the requirements in a more conventional way through a use case diagram. Each use case specified in the diagram will be described in more detail in the following subsections. This diagram shows the two kinds of user, administrator and guest. An administrator inherits all the functions of a guest.

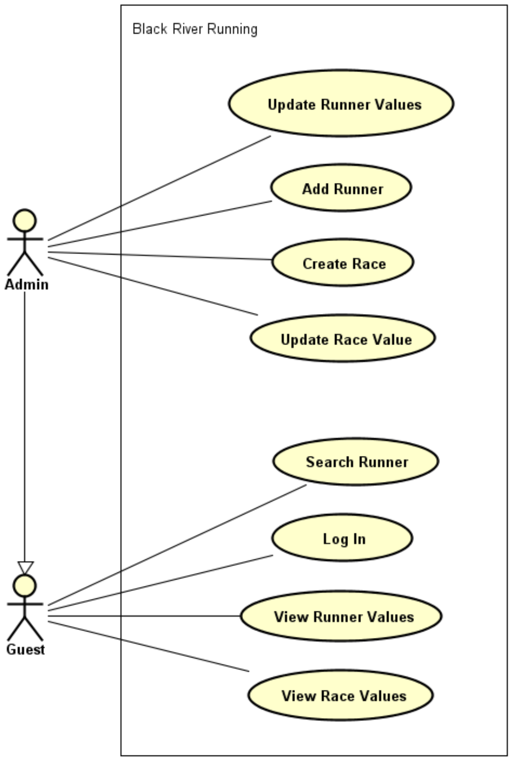

Figure 2: Use Case Diagram.

8.1 Log In

- **Initiator :** Guest.
- **Goal :** Log in to admin page.
- **Main Flow Of Events :**
  1. Guest enters login information (username and password) in the fields.
  2. System checks validity of the information.
  3. Guest is logged in.
- **Alternate Flows Of Events :**
  - 1 : Information are not valid.
    1. System shows an error message.
    2. Resume at 1.

8.2 View Race Values

- **Initiator :** Guest.
- **Goal :** View data of a specific race.
- **Main Flow Of Events :**
  1. Guest selects a race.
  2. System displays race data.

- **Alternate Flows Of Events :**
  - 1 : The selected race is not valid.
    1. Stop the case.

8.3 View Runner Values

- **Initiator :** Guest.
- **Goal :** View stats of a specific runner.
- **Main Flow Of Events :**
  1. Guest selects a runner.
  2. System displays runner stats.
- **Alternate Flows Of Events :**
  - 1 : The selected runner is not valid.
    1. Stop the case.

8.4 Search Runner

- **Initiator :** Guest.
- **Goal :** Search for a runner.
- **Main Flow Of Events :**
  1. Guest enters runner information.
  2. System displays runners that match this information.

8.5 Update Race Values

- **Initiator :** Administrator.
- **Goal :** Update race values.
- **Main Flow Of Events :**
  1. Administrator selects a race.
  2. Administrator updates race values.
- **Alternate Flows Of Events :**
  - 1 : The selected race is not valid.
    1. System shows an error message.
    2. Resume at 1.

8.6 Update Runner Values

- **Initiator :** Administrator.
- **Goal :** Update runner values.
- **Main Flow Of Events :**
  1. Administrator selects a runner.
  2. Administrator updates runner values.
- **Alternate Flows Of Events :**
  - 1 : The selected runner is not valid.
    1. System shows an error message.
    2. Resume at 1.

8.7 Add Runner

- **Initiator :** Administrator.
- **Goal :** Add a runner to race.
- **Main Flow Of Events :**
  1. Administrator selects a race.
  2. Administrator choose either an existing runner or a new runner to add to the race.
  3. System add the runner to the race.
  4. System sends validation message.
- **Alternate Flows Of Events :**
  - 2 : Administrator choses existing runner.
    1. Administrator enters runner SSN (Social Security Number).
    2. Resume at 3.
  - 2 : Administrator choses new runner.
    1. Administrator enters runner information.
    2. Resume at 3.
  - 1 : The selected race is not valid.
    1. System shows an error message.
    2. Resume at 1.
  - 2 : The selected runner is not valid.
    1. System shows an error message.
    2. Resume at 1.

8.8 Create Race

- **Initiator :** Administrator.
- **Goal :** Add a race.
- **Main Flow Of Events :**
  1. Administrator enters race information.
  2. System add the race.
  3. System sends validation message.

9 Time Plan

In this part of the report, the focus is on the time aspect of the project in order to get a better understanding of it.

First the deliverables table shows the project artefacts associated with their deadlines.
Then, a Gantt Chart shows the activities planning for the project.
This part is ended by the planned working hours table, that shows working hours for each members of the group.

9.1 Deliverables

The following table shows the deliverables deadlines fixed for this project.

Title	**First version**	**Final version**
Project Plan		2018/11/22
Detailed Design Description	2018/12/06	2019/01/17
Product	2018/12/06	2019/01/17
Project Report		2019/01/17

9.2 Gantt Chart

This subsection shows the Gantt Chart that the group wants to follow throughout the development period. On the left, there are all the activities to complete in order to finish the project. At the top, there is the period of work.

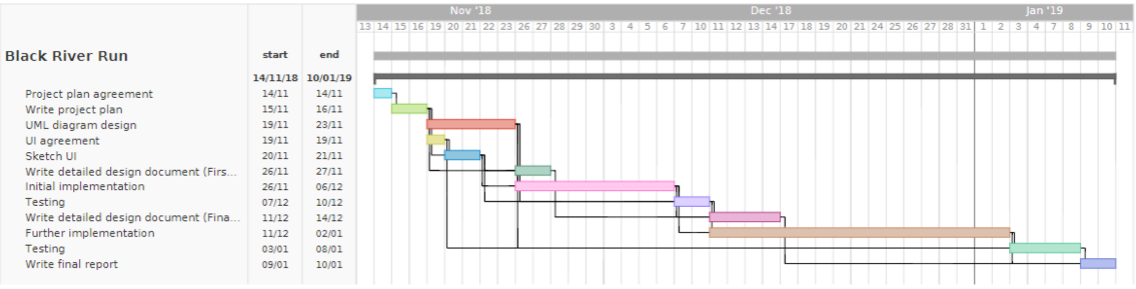

Figure 3: Gantt Chart

9.3 Planned Working Hours

Each week, the members of the group have to meet for 5 to 10 hours. Furthermore, weekly presentations take about 30 minutes. Each member also provides a personal work that he/she shares using the GitHub directory. The total of this working hours is specified in the following table.

Member	W46	W47	W48	W49	W50	W51	W52	W1	W2	W3	Total
Bastien	12	16	20	18	15	15	5	8	10	8	127
Zacharias	12	16	20	18	15	15	5	8	10	8	127
Sebastian	12	16	20	18	15	15	5	8	10	8	127
Cécile	12	16	20	18	15	15	5	8	10	8	127
Mohammed	12	16	20	18	15	15	5	8	10	8	127
Rikard	12	16	20	18	15	15	5	8	10	8	127
Johannes	5	16	20	18	15	15	5	8	10	8	120

10 Development

10.1 Overall Functionalities

The main functionality of the system is to keep track of runners in Black River Run organized by Västerås Löparklubb. Every runner carries a device for time keeping. The data from the device is collected by the system and is displayed on the webpage.

The system includes an administration page where the race organizers can update data about the runners and races. From the administration page, the physical time tracking units can be swapped and time-stamps can be modified. This provides a solution for fixing faulty units during a race.

10.2 Existing System

The runners carry with them a SI-Unit, that makes it possible to monitor their movements during the race. When the race starts, the first punch is sent to the third-party server. The information is later retrieved and stored in the database, which in turn is forwarded to the webpage. The current webpage is written in PHP and hosted by Loopia.

10.3 Constraints

There is a requirement that the back-end needs to be written in either PHP, Python or Perl because the program is run on a shared webhosting service called Loopia, which only supports the mentioned programming languages. There are no other restraints on other parts of the system. However, the client suggested that Angular could be used for the front-end as that is a frame-work that he sees more and more on the job market.

10.4 Initial Backlog

This subsection presents the initial backlog of the project. It is the set of tasks, determined so far, to achieve in order to produce a result corresponding to the client's expectations. A task is represented by its name, its description and its priority over other tasks.

Name	**Description**	**Priority**
1.0 Create Database	Store data.	Critical
1.1 Create Stored Procedures	Simplify fetching and storing data.	Low
1.2 Create Database Schema	Mapping the database structure.	Critical
2.0 Get Punches	Fetch data from the API.	Critical
2.1 Format Punch Data	Format the data from GetPunches.	Critical
2.2 Store Formated Punch Data	Store the formated data in database.	Critical
3.0 Fetch Runner And Race Data	Fetch data from the database.	Critical
3.1 Update UI With Runner And Race Data	Update the guest website with the data.	Critical
3.2 Estimate Runners Arrival Time At Stations	Estimate when a runner will arrive.	Medium
4.0 Login	Login to admin page.	High
4.1 Add/Update Runner	Add/update a runner.	Medium
4.2 Add/Update Race	Add/update a race.	Medium
4.3 Correlate Runner To Race And SI-unit	Correlate Runner to Race and SI-unit	Medium
4.4 Update Timestamps	Manually update timestamps in the database.	Low
5.0 View All Runners	List runners of a race.	Medium
5.1 View All Races	List races.	Medium
5.2 View Individual Runner	List statistics of an individual runner.	Low
5.3 View Individual Races	List runners/statistics of an individual race.	Medium
5.4 Map of Active Runners	Create a map and show where runners are.	Very-Low
6.0 Test System	Test all parts.	High