Tampere University Unit of Computing Sciences
COMP.SE.610 Software Engineering Project 1
COMP.SE.620 Software Engineering Project 2

Hive/Group-01

PROJ-S2023-G01-Geocaching-front-end

Final Report

Andrey Essaulov
Juho Keto-Tokoi
Melany Mendoza Romero
Harri Hietanen
Nelli Leinonen
Mikko Kaukonen
Matias Kaakinen

Version history

Version	Date	Author	Description
1.0	29.04	Andrey Essaulov	Filling basic project information
2.0	03.05	Andrey Essaulov	Chapters 3 and 4
3.0	04.05	Andrey Essaulov	Rest of the document

Contents

1 Introduction

1.1 Purpose of the report

Purpose of this report is to document the state of the Geocaching front-end project. This document reflects the current state of the project, processes, requirements, teamwork and testing. This document is reflecting how work on the project was done and team results.

1.2 Product and environment

The product is a Geocaching website. The website contains all the information about geocache objects placed somewhere in the real world with coordinates of these objects. End-users can try to find objects or add new objects to the website.

The customer's owned website, Geocaching.fi, is partially integrated with Geocaching.com.

It is possible to buy premium subscription for account to have additional features.

Users of the website are geocaching hobbyists, nature tourists and generally people that like to walk outside.

Currently it is possible to open a website on the phone, but there is now extensive support for doing it. General experience of using a website with a phone is not good enough.

Look-a-like systems: Geocaching.com solutions, map projects such as google map.

1.3 Definitions, abbreviations and acronyms

- Customer - owner of the business
- End-user - real, non-team-member person, that will used service before or used analogue services
- Geocaching - outdoor activity about searching and creating caches - small objects in the real world.
- VCS - Version control system

2 Project organisation

2.1 Group

Name	Contact	Role	Team
Juho Keto-Tokoi	juho.keto-tokoi@tuni.fi	Senior Software Developer	Software Development Team
Melany Mendoza Romero	melany.mendozaromero@tuni.fi	Software Developer	
Harri Hietanen	harri.hietanen@tuni.fi	Software Developer	

Nelli Leinonen	nelli.leinonen@tuni.fi	UX/UI Designer	UX/UI Team
Mikko Kaukonen	mikko.kaukonen@tuni.fi	UX/UI Designer	
Matias Kaakinen	matias.kaakinen@tuni.fi	QA Manager	QA Team
Andrey Essaulov	andrey.essaulov@tuni.fi	Project Manager; Product Owner	Management

Next table is a processed survey on team Experience with different technologies and role preferences:

Name	Experience	Preferences
Juho Keto-Tokoi	Frontend: Javascript/Typescript, React Backend: Nodejs + express, Kotlin + Spring boot Database: Postgresql, MongoDB Other: Python, C++	Senior Software Developer
Melany Mendoza Romero	JavaScript, React, Java, PHP (a little), Python, MySQL	Software Developer and/or UX/UI (not much experience only from uni courses)
Harri Hietanen	C++, Python, Java, Javascript, PHP	Software Developer
Nelli Leinonen	JavaScript, TypeScript, Node, React, C#, some C++ and Python from university course projects	UI/UX, Software Developer
Mikko Kaukonen	C++ and Python, a little bit of javascript, html and css.	Software Developer / UI/UX(not so much experience but ready to learn) / QA(not so much experience but ready to learn)
Matias Kaakinen	C#, ASP.NET (core), JavaScript/jQuery, Node, React (only little), CSS, SQL, C++, PHP (barely), Java (early studies), Python (early studies)	Software Developer (Full Stack experience), QA (interested, less work experience), UI/UX (interested, less work experience)
Andrey Essaulov	C#, C++, Python, ASP.NET core , javascript, html, css	Manager, QA

2.2 Customer

Name	Contact	Role
Harri Hirvasniemi	harri.hirvasniemi@kivenalla.fi	Business Owner (Customer)
Jussi Kallioniemi	jussi.kallioniemi@radiant.fi	Business Technical Specialist

2.3 Other Stakeholders

Name	Contact	Role
Timo Poranen	timo.poranen@tuni.fi	Tampere University Coach
Geocaching.com	Geocaching.com	Owner of Geocaching original API
Weellu	velipekka.eloranta@gmail.com	Geocacher that agreed to be end-user for testing

3 Project Implementation

3.1 Communication

Communication is split into 2 main categories. Inner development team communication and Outer communication. Inner communication is used by development team members for keeping in touch with each other and organize work. Outer communication is organised between development team, course staff and Customer. For all these types of communication are used different tools, that fits nature of the communication

Inner communication is fast, mostly non-formal and requires immediate attention from the team. Therefore, those tools were picked:

- Telegram – initial instrument for team organization. Convenient to use everywhere and it is easy to acknowledge notification.
- Discord – core communication for team, with separate chats and tags system it is a perfect tool for communication in teams.

Outer communication is mostly formal and requires a lot of planning between different sides to organize meeting or receive requirements. Therefore, email was picked as main communication provider.

3.2 Tool and Technologies

Our main choice of different technologies is motivated by several reasons, but the main one is competence of team members in this tool. There were picked programming stack, design tools, version control, CI/CD pipeline.

- **Programming stack:** React (v18.2.0), Typescript (v4.9.4)
- **Design tools:** Figma
- **Communication:** Discord, Telegram
- **Version control server:** Github
- **Kanban Board:** Github Projects
- **Documentation:** Google drive
- **Working time tracker:** MMT
- **Calendar:** Google calendar

3.3 Sprints

Sprints are 2 weeks periods that used to check retrospectively what have been done and decide what should been done next.

Course provides its own calendar, but it starts a bit later, then our team has started work. Also, course sprint calendar contains some inconsistency in sprint periods. Therefore we have created our own calendar with keeping in mind original sprint numbering.

Here we provide our calendar and real content that was done in this timespan

Sprint №	Dates	High level content
Essential	10.01 - 22.01	Team assemble. Role assignment. First contact with Customer. Requirements gathering. Tools initialization.
0	23.01 - 05.02	Process and update requirements from Customer. Initial implementation of essential features. First iteration on UI design
1	06.02 - 19.02	Map working prototype, first design of the main page, CI/CD pipeline setup, Writing QA test plan
2	20.02 - 05.03	Authentication, Design iterations on maps and authentication,
3	06.03 - 19.03	Cache implementation, Design cache page, QA testing sessions
4	20.03 - 02.04	Feature Development. Design iterations, QA testing sessions
5	03.04 - 16.04	Filters implementation, bug fixing, QA testing sessions
6	17.04 - 30.04	Project finalisation. Writing documentation on project exploitation. Customer testing, approval. Final end-user testing

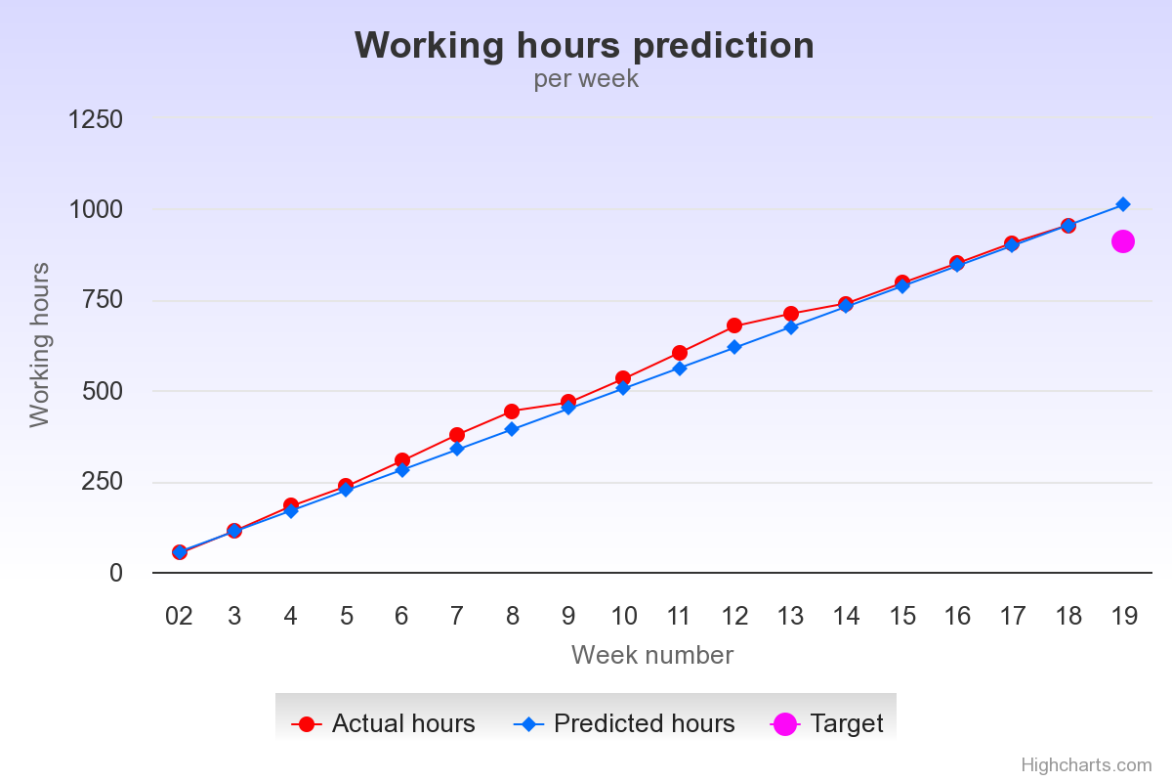
Product Backlog has been constantly updated through the process of the development. Here are some statistics related to the backlog.

- Completed tasks – 92
- Backlog tasks left – 22

Next is information about which sprint what amount of tasks has been done. We have not tracked initial amount of tasks in the start of the sprint, therefore some tasks can be assigned at one sprint and move to next.

Sprint	Tickets done
1	18
2	10
3	15
4	13
5	16
6	20

Task prioritization was introduced through tags on Kanban board on late stage of development. It was a mistake, and we should introduce it earlier because it made development so much easier from production point of view. However, in the late stage it became really handy tool, because we had several type of tasks that we should handle. Also, it became obvious that we can not make everything required and we need to pick some tasks as high priority.



3.4 Deliverables and outcomes

Deliverable	Description
Project Poster	PDF one page document with visual representation of the project nature
Test Repost	Results of testing process through project lifespan
Final Report	Final document with project description
Source code	Code itself that represents project
Technical documentation	Documentation contains technical specifics that are described the way that it is possible to connect back-end to front-end
Design document	Figma file that contains all design decisions that were approved by Customer and created by design team.

Product itself is React front-end project that have can make requests and process responses from back-end.

To make project work it is required to return for end-user new front-end version in case it is a mobile device and make back-end support all requests and front-end can send. More specifically about API that front-end supports is possible to read in Technical Documentation.

Our team has not tracked all the data related to amount of classes and methods, but this is our approximation:

Self-coded lines	3607
Number of classes	80
Number of methods	200
Number of views	7

Worth to mention here that we have not reused code from the current project of Customer. This why we do not have graphs related to reused code.

3.5 Restrictions and limitations

Main restriction for this project is that it was created mobile-first. It should work for mobile phones mainly and it is not guaranteed quality work on the desktop devices with bigger screens.

All other limitations have more technical point of view and based on points of interaction between front- and back- end.

For more information consider looking Technical documentation.

3.6 Third party components, licenses and IPRs

The only third party component, except programming technologies is OpenStreet map api to show map to users.

3.7 GDPR, ePrivacy and related matters

Project does not keep any personal information of user. Therefore from front-end point of view do not happen any legal related questions.

All personal information is held in the server, which already regulated all of those questions.

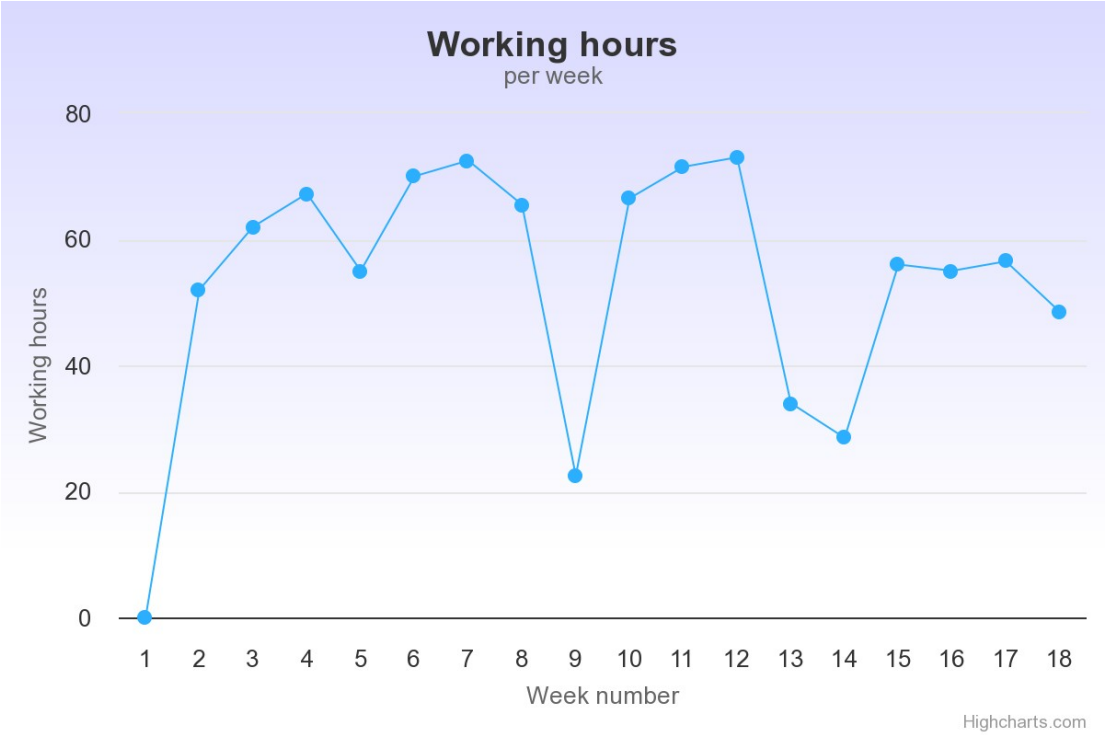
4 Working Hours

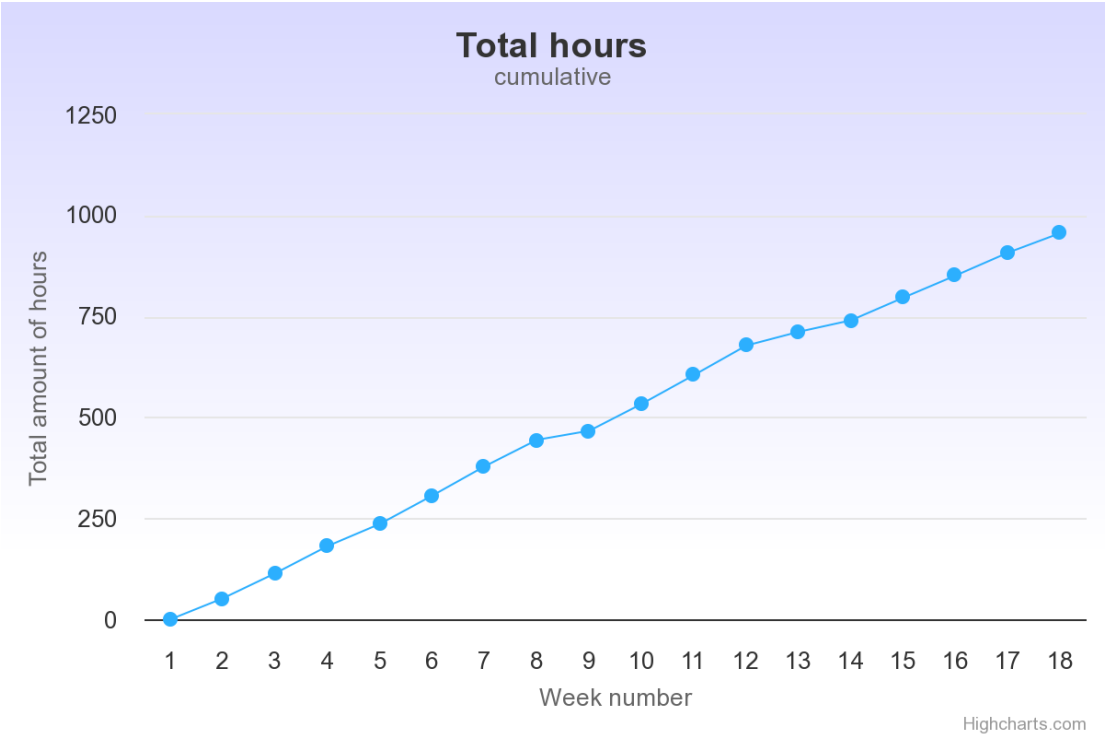
Table that contains information about team member spent hours divided by MMT categories.

	Andrey	Matias	Melany	Nelli	Harri	Juho
Documentation	37.5	54	0	7.5	0	12.5
Requirements	8.8	1	0	3	0	2
Design	0	0	0	64.5	0	0
Implementation	0	0	49.5	0	32	61
Testing	6.5	15.5	6	2.5	4	11
Meetings	25.5	27.5	28.5	26.5	35	31.5
Studying	2	27.5	31	10.5	44	4
Other	41.2	0	6	1	4	11.5
Lectures	10	6.75	11	17.5	22	13
Total	131.5	132.25	132	133	141	146.5

Next table is hours spent per week of work

Week 1	-	Week 10	67
Week 2	52	Week 11	72
Week 3	62	Week 12	73
Week 4	67	Week 13	34
Week 5	55	Week 14	29
Week 6	70	Week 15	56
Week 7	73	Week 16	55
Week 8	66	Week 17	57
Week 9	23	Week 18	49
Total Hours		**956**	





Our team has not collected weekly hour promises in project plan, so we can not reflect on how all of those number has changed. Also, we lack in information how much time has been spent per task. This information can be be obtained manually from MMT, but it does not provide any convenient interface to collect this information and present it.

4.1 Personal contribution

Juho was responsible for all software team and project architecture. Has created a core for the project implemented a most crucial map features, created mock back-end, has written technical documentation, has guided his team members, was an active member of the team.

Harri was responsible for authentication features in the project and several filters implementation. He has taken responsibility for several features for they full lifespan: planning, development, communication between UI team, implementation, writing tests. Active member of the team activities.

Melany was responsible for some most of the filters and cache implementation. She has full working pipeline for those features and was responsible for all the results. Active member of the team communication.

Mikko is designer in UX/UI team that was responsible for design of search page, navigation bar design. Communicated with Customer and processed feedback from they.

Nelli is main responsible for Figma maintaince. Designer of project poster, cache page, login page, map filters and many other design tasks. Communicated with Customer and processed feedback.

Matias is QA person in the project. Responsible for quality of the product itself. Defined and tested everything related to quality of the final product. Implemented several documents and maintained test report.

Andrey manager of the team. Main communicator between course staff and Customer. Regulated tasks in the sprints, defined what should be done in the current sprint. Tracked team progress and made reports to the Customer.

4.2 Peer feedback

We have to utilize peer feedback review tool. It was nice to receive those comments about your work. It is hard to say did it help to improve our performance, because we have never discussed those result with each other. If there were some changes in performance, then they were almost untraceable.

This tool is okay, helps to stop and think about performance of different team members. Only thought is that it is possible that some team members do not contact very closely with each other and there can be nothing to say about them.

5 Quality Assurance

5.1 General Description of testing

Main idea of out testing process is that we want to check all our functionality with in a limited amount of time. We have special person to do it, Matias, who is representing QA team.

To achieve maximum quality that we can afford we have decided to use mix of automotive testing and exploratory testing. Unit tests are written by developers and exploratory tests done by QA and project managers.

We wanted to make not-biased usability testing sessions with real end-user, but we failed to organize it in time.

All testing documentation is handled by QA manager with testing diaries and testing reports.

5.2 Bug Reporting

All existing functionality does not contain critical bugs in it. All bugs that we know about right know have only minor impact. In all expected cases front-end works correctly, that is proved by testing sessions.

Overall, were found 12 bugs of different level of crucially and impact. 10 of them are fixed, 1 – is about our CI/CD pipeline and does not affect production. The only bug left is “center button can take a lot of time”.

5.3 Conclusions on product’s quality

Product currently is bug-free and we can expect that there will no be critical bugs in production. All testing measures that we have applied – showed us that current state is production viable. It is still worth to check additionally on how end-user will interact with application.

6 Risks and Problems

6.1 Foreseen Risks

Our team has been expecting next risks:

Risk ID	Explanation
0	Team member cannot continue to work on the project
1	Customer does not replying on the email
2	Health accidents
3	Team achieved 130 hours of work per team member, but project is not ready yet

All of those risks are not realized, so they left only as risks. Some of the risks not even were update in MMT, because they are so low level of possibility to happen. We have not updated this list since we bran stormed it and this what is possible to improve in the future. Once in awhile gather with team and talk about risks that they can think of in current stage of development.

6.2 Risks not foreseen

There were 2 main risks that happened to us. The first one we failed to handle end-user testing. We expected to start do it those test in the middle of development, however we have postponed start of those testing sessions 2 times. Then, when we have decided that we are ready to show something working to Customer, we realized that we do not have infrastructure to do it. Several weeks we have been receiving access to Customers server, so we can deploy our application here for demonstration. And when we contacted end-user, we have not received a reply. Our end-user testing has not happened after all.

The second risk is that we have not be able to finish whole backlog in those 4 months. It was pretty reasonable in the start of the project that we will finish core functionality in advance and then will start add some additional features. However, we have done most of the requirements, but some of them still left unimplemented.

7 Not implemented in this project

7.1 Rejected ideas

Main rejected idea is full scale Test-Driven Development. We have implemented this technique only partially in development, but experience of developers and amount of work required made this unsustainable for us. In other words, we have declined testing everything and writing tests first, because it would take too much time, that can be redirected into feature implementation.

7.2 Further development

Currently all further development is in the first place is finishing not-implemented functionality.

- Cross-hair in the centre of the map
- Footer with additional information
- Finish all filters
- Advertisement banner
- Switch between map types
- Credits page
- Authentication: forgot password button

All of those are currently in backlog, but has not been implemented.

8 Lessons learnt

We have started early in the project and it was a good decision and it helped a lot with taking a good pace of development.

Our main development driver was our senior software developer. Most experienced developer is really running everything further.

Team meetings are very important point of synchronization between team members. It is better when there is more of them, however 1 per week is still good enough.

Discussing problems in the development process is important, because tracking problems as early as possible can help to reduce overhead work in the future.

Picking technological stack that is main developer familiar with is a win-win situation, because it reduce time for head start. Other developers can start learning new tech under control of already experienced developer.

Choosing people that are specifically work on the design is a good idea if you need to provide a lot of new designs.

There should final decision maker, that can make final choice in case there is no agreement in team.

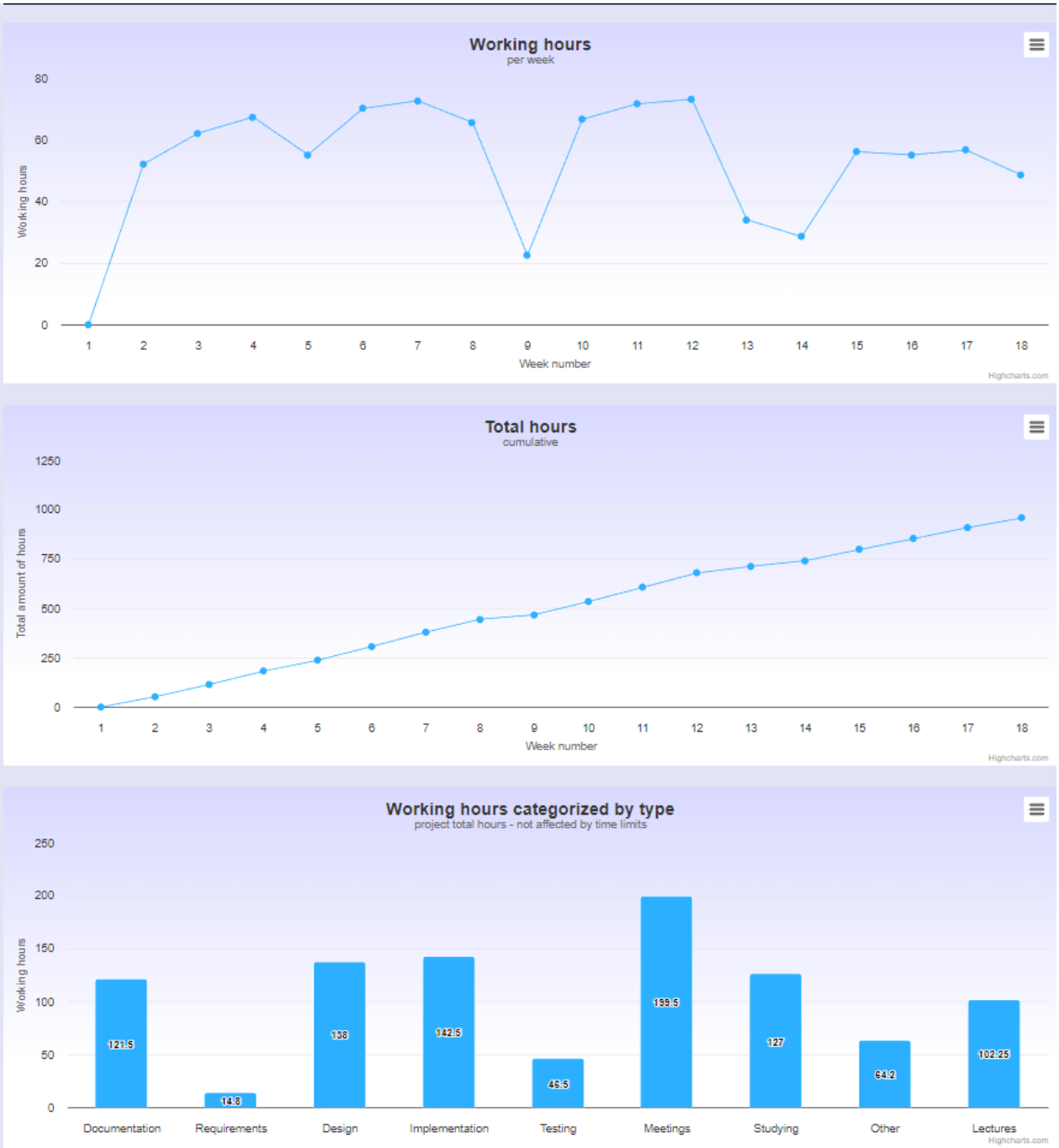
9 Comments about the course

This course has great organization. Every step of the course is clear and understandable. Assigned coach is good idea for communication between team and course staff. Review meetings with Customer and Coach helps to keep team on track. All events are greatly handled. All services related to the course work greatly, no technical issues.

The course moodle page is a little bit overwhelming on information. It can be sometimes hard to find something that your looking, because there is a lot of stuff. Course presentation contains a lot of repetition of information. Course idea to provide real world experience is great, but if student is working already, then this course can not provide anything view – maybe, working students should be able to pass this course through learning diaries that they fill on work.

10 Statistics

Geocahing Front-End, Hive Team, 7 people	
Customer	Harri Hirvasniemi Geocache.fi
Tools	React, Typescript, Figma, Discord, Telegram, Github, Github Projects, Google drive, MMT, Google calendar
Lines of Code	3607
Number of classes	80
Version control revisions	195
Implemented tickets	92
Not-implemented tickets	22
Personal thoughts	Greatest team ever existed!


Working hours
per week
Working hours
Week number
Highcharts.com
Total hours
cumulative
Total amount of hours
Week number
Highcharts.com
Working hours categorized by type
project total hours - not affected by time limits
Working hours
121.5
14.8
138
142.5
46.5
199.5
127
64.2
102.25
Documentation
Requirements
Design
Implementation
Testing
Meetings
Studying
Other
Lectures
Highcharts.com