Project Plan
Energy Grid System

Date	23/02/2022
Version	1.0
State	WIP
Author	The Development team[1]

[1] The Development team consists of all team members (Aleksandar Popov, Alexander Tsvetkov, Dima Ratuşniuc, Dimitar Ivanov, Georgi Minchev, Kristian Lachev, Velimir Vukašinović)

Version history

Version	Date	Author(s)	Changes	State
0.1	08/02/2022	The Development Team	Creating the document	WIP
1.0	23/02/2022	The Development Team	First version of Project Plan	Done

Distribution

Version	Date	Receivers

Table of Contents

1. Project assignment

1.1 Context

With the current Dutch energy sector, the electricity supply monitoring is limited to only between providers and consumers, which includes only the commercial electricity. The growth of self-produced electricity (renewable electricity) brings the question on how to avoid both under-capacity or over-capacity of an electrical grid, which are linked with additional costs.
Infiniot is a software consulting Dutch company for grid operators, trying to find a solution for this problem.

1.2 Goal of the project

In order to avoid blackouts or overloads, Dutch energy providers, as well as consumers, should be in constant connection between each other. This way simulations can be run, which will help providers to set up the electrical grids in advance. The predictions made will be based on the user data, which they will be provided with, as well as certain factors that are most often easily predictable, such as weather conditions, location, timestamp in year/day/hour, etc.
Being a user, the priority of the information differs. People should be aware of their electricity usage and how much this would cost them, as well as their production (if there is any). Users can also benefit from a certain simulation, as they will be notified in advance of an estimated price and consumption, which will help them plan ahead.

Infiniot asked our group to create software solutions for avoiding the problems caused by an overload and underload, as well as keeping users and providers informed with estimates and current usage of the system.
Both of those problems can be solved by making a connection between the consumer and producer with the help of a web application for energy companies and mobile application for all the users.
Due to the simulations that will be run, a monitor app will be useful, that will give an overview of the load created on the device.
The main focus of our group will be in the provinces of Noord-Brabant, Limburg, Gelderland and Zeeland.

1.3 Scope

Inside scope:	Outside scope:
Web application - providing utility companies with a tool which can show various energy-related statistics, information about the energy grids in the company's region and forecast future energy usage and consumption. Regional map with all the power grids. Present data in the form of diagrams and charts. View historical data as well as predicted data using our prediction model. Different types of data can be viewed, such as monthly, weekly, daily and hourly values. Possibility to generate a PDF report for each given hour/day/month/year. Possibility for an admin to switch to the Monitoring Application straight from the Web Application.	Manual for the application will not be provided.
Monitoring tool - providing technical support. Collects, stores, and analyzes data and metadata necessary for troubleshooting and optimizing performance. Tracks - Error rates, CPU usage, Response times, Request rates, Uptime, Number of instances.	There will be no training provided to use the application.
Mobile application - informing the consumer about the current usage of energy in the household and alerting if there is a chance of potential overload. Provide in depth information about the consumption, costs, production and earnings of energy in the form of graphs. Predict future consumption, costs, production and earnings of energy taking certain variables into account, such as weather, season, price, time, past usage, etc. User-friendly design. Not overwhelming the user with complicated information.	Bugs found after the final phase will not be fixed unless an extra contract is added.
Source code of all applications	Design changes after the final phase will not be done by the development team unless an extra contract is added.

1.4 Strategy

In this project we will be using the Agile approach which means we are concentrating on frequently delivering working software. We decided to choose Scrum framework since this method assumes that our client might change the requirements or add new to the existing ones during the developing process.

Each week we conduct at least 2 stand up team meetings in which we discuss what everyone has done since the previous one. The team consists of 7 members who are equally working on delivering the product features demanded by the client within all the 6 iterations.

1.5 Research questions and methodology

To explain and structure our research we will be utilizing the DOT (Development Oriented Triangulation) framework which will help us to combine the appropriate research methods and strategies to design a research pattern for answering the research questions that come up during the project realization.

The research questions defined during the **Hackathon** week (Week 1) of the project are listed in the next paragraph of this Chapter. Apart from that we have created a separate Research Document, dedicated to providing a comprehensive view on the research strategies and research questions we come up with in the next phases of the project. The answers of the research questions listed below are also documented in the Research Document.

Hackathon week research questions

After analyzing the project assignment description the following research questions have been defined:

- **How to determine the main requirements of the different actors/stakeholders?**
- **Where to find relevant data and statistics to use when making predictions in the energy domain?**
- **Taking into account each stakeholder, should the functionality be split into multiple applications or not?**

  After we gathered enough information from the assignment description and from the project owners we decided to split the system into multiple applications based on the different users' requirements:

  - Web application for the big consumers and utility companies

- Monitoring tool for the technical support
- Mobile application for smaller consumers as households

After we split the applications more research questions came up. The questions are as follows:

Web Application research questions:

- **What will the Web Application be about?**
- **Who will benefit from this Web Application?**
- **What is the main goal of this application?**

Monitoring Application research questions:

- **What is an Application monitoring tool and what metrics should it display?**
- **How to visualize data to make it easier to understand and process?**
- **What events should be visible in the logs of each service?**

Mobile Application research questions:

- **What information the consumer would like to see in a mobile application?**
- **How to predict the consumption, costs, production and earnings?**
- **How to present the information in a user-friendly way?**

- **What tech stack are we going to use?**
- **How are we going to test the system?**

1.6 End products

The end products of this project are:

- **Web Application**, capable of showcasing data about electrical grids in the region, with the data being showcased both for the past and present, as well as simulated data for the future;

- **Mobile application**, showcasing data about electrical usage and production of individual clients;

- **Backend API**, which will process data, supply it to the Web and Mobile app and tackle logic related to the application in full;

- **Documentation about all the research** done in order to answer the research questions of this project;

- **Design documents**, where the structural elements such as Architecture and the UI designs are detailed and justified;

- **General documentation**, where documents about the requirements, the analysis done, etc. are gathered.

The Project Breakdown Structure Diagram can be found on the following link.

2. Project organization

2.1 Stakeholders and team members

Name	Role and functions	Responsibilities	Availability
Kenneth Ruys (kruys@infiniot.nl)	*Stakeholder and a contact person from Infiniot*	-	*Available for questions via email.*
Ton Smets (tsmets@infiniot.nl)	*Stakeholder and a contact person from Infiniot*	-	*Available for questions via email.*
Gertjan Schouten (gertjan.schouten @fontys.nl)	*Stakeholder, product owner and mentor of our group.*	-	*Available on Thursdays, by MS Teams and email.*
Marcel Boelaars (m.boelaars@fonty s.nl)	*Stakeholder and a technical teacher*	-	*Available on Mondays, by MS Teams and email.*
Tom Meulensteen (t.meulensteen@fo ntys.nl)	*Stakeholder and a technical teacher*	-	*Available on Mondays, by MS Teams and email.*
Velimir Vukašinović (v.vukashinovich@s tudent.fontys.nl)	*Team leader and full-stack developer*	*Ensures everybody works on their assigned task.*	*Available from 09:00 to 16:00 throughout the weekdays, by email or after scheduling a meeting beforehand.*
Aleksandar Popov (427643@student.f ontys.nl)	*Git master and full-stack developer*	*Manages Gitlab Boards, creates issues and assigns tasks, reviews code*	*Available from 09:00 to 16:00 c throughout the weekdays, by email or after scheduling a meeting beforehand.*
Georgi Minchev (g.minchev@stude nt.fontys.nl)	*Scrum master and full-stack developer*	*Schedules team meetings and interacts with POs, takes minutes of the meetings*	*Available from 09:00 to 16:00 throughout the weekdays, by email or after scheduling a meeting beforehand.*

Alexander Tsvetkov (a.tsvetkov@student.fontys.nl)	*Full-stack developer*	*Development of the mobile application, full stack development*	*Available from 09:00 to 16:00 throughout the weekdays, by email or after scheduling a meeting beforehand.*
Dima Ratuşniuc (d.ratusniuc@student.fontys.nl)	*Full-stack developer*	*Docker and Kubernetes deployment, backend development*	*Available from 09:00 to 16:00 throughout the weekdays, by email or after scheduling a meeting beforehand.*
Dimitar Ivanov (dimitar.ivanov@student.fontys.nl)	*Full-stack developer*	*Development of the mobile application, front-end development*	*Available from 09:00 to 16:00 throughout the weekdays, by email or after scheduling a meeting beforehand.*
Kristian Lachev (kristianlachev00@gmail.com)	*Full-stack developer*	*Development of the web application, full stack-development*	*Available from 09:00 to 16:00 throughout the weekdays, by email or after scheduling a meeting beforehand.*

2.2 Communication

For communication between the team members we have created a Discord server with different channels (voice and text) in order to organize our work better (questions, announcements, meetings, etc.). Each week we organize two group meetings to synchronize with each other what everyone has done. Most of our group meetings take place online because of the current situation with COVID-19.

We also have a separate channel (in Microsoft Teams) in which we can directly approach our coach for further questions or to set up meetings. Each Thursday we organize a weekly meeting with the coach and the company stakeholders in order to report our progress.

3. Activities and time plan

3.1 Phases of the project

The project is going to be split in 5 sprints with each being 3 weeks long. During the first week the team is going to focus on sprint-planning while also doing research on the topics relevant to the planned activities. The second week is all about putting the knowledge, gathered during the research, in action, by implementing features, creating designs or working on important pieces of documentation. In the final week of each sprint, we are going to test the new features and prepare the sprint demo. At the end of the sprint, the team is going to present the progress that was made during the sprint. Then we are going to hold a retrospective meeting where we discuss the results of the sprint and feedback received from stakeholders.

3.2 Time plan and milestones

Phasing	Planned tasks	Effort	Start date	Finish date
Sprint 0	Project understanding, general planning, prototyping, setting up the work environment	3 weeks	07-02-2022	27-02-2022
Sprint 1	Backend implementation, designing frontend for web app and mobile app, data preparation for prediction model.	3 weeks	07-03-2022	27-03-2022
Sprint 2	Web app frontend, integrating backend and frontend, AI model for predictions	3 weeks	29-03-2022	18-04-2022
Sprint 3	Start with mobile app, monitoring tool implementation, AI implementation.	3 weeks	19-04-2022	09-05-2022
Sprint 4	Testing of the web application and the mobile application	3 weeks	10-05-2022	30-05-2022
Sprint 5	Fixing bugs and delivering.	3 weeks	31-05-2022	20-06-2022

4. Testing strategy and configuration management

4.1 Testing strategy

Backend
During the project we will make use of various testing methods. The testing methods will be separated into two parts - **functional** and **non-functional**. Of the functional tests we will be mainly making use of **unit tests**. Unit testing is a type of testing where individual units or components of the software are tested. The tests are automatic and having unit tests inside a system means that finding bugs would be easier in the future. We will also be making use of **acceptance testing**, which is a testing technique that determines whether or not a software system completes the requirements set by the stakeholders. We are also going to do **integration testing,** which are tests that automate both server and frontend testing**.**

Frontend
Of the non-functional tests we will be making use of **performance tests**, **security tests** and **usability testing**. Performance tests can be done using Google Lighthouse, which gives a good overview of the performance and supplies a lot of data, which can help developers improve performance. Security testing is mandatory in every project in order to ensure the stability of the project during an online attack of some sorts, it mainly serves to reveal security flaws in the system. Usability testing is another type of test, where a new user is presented the system and given certain tasks to complete using the system, without any help from the developers. This is done in order to improve the usability of the system interface and gauging the user experience.

Code Quality
For the code quality testing we will use **SonarQube,** which is an open-source platform for code quality inspection. It will greatly reduce the time needed to go through all the code to find and fix code quality inconsistencies.

4.2 Configuration management

We are going to be using GIT as a version management system during the process. Alongside, we will keep a backlog of issues and tasks using Gitlab Boards. The Git master is the person responsible for creating issues and assigning people to them.

We decided to use separate branches for each new feature or issue that we have. After the feature is completed or the issue is resolved the code is going to be reviewed by a member of the team and then merged into the development (dev) branch. Before the presentation at the end of each sprint a

stable version of the application is going to be pushed to the release branch. If needed the application can be reverted to a stable version.

To make the development process easier, all components of the system are going to be deployed and run in Docker containers. Using dockers Dev Environments using docker would allow us to easily set up repeatable development environments, keeping the environment details versioned along with the code code. We can also switch between team members' environments, move between branches to look at changes that are in progress, without moving off the current Git branch. Work-in-progress code can be shared in just one click and without having to deal with any merge conflicts. The containers are spun up when a team-member is working on the system. By doing this, we ensure that every part of the system is running during development.

Sensitive data such as database passwords and API keys is going to be stored in a .env file. This file provides the necessary environment variables to run the docker containers for each specific application that is part of the system.

5. Risk and mitigation

Risk	Impact	Probability	Prevention
Possible miscommunications within the team	High	Low	We make sure to discuss every decision within the team with every member of the team.
Possible miscommunications with the client or tutor	High	Low	We keep track of every meeting by using Agendas, before the meeting to notify about it and the topics we wish to discuss during it, as well as Minutes of the meeting, to make sure everything said within it is remembered as it is.
Team members might get sick	Medium to high	Medium	We have taken steps in our scheduling to account for unforeseen circumstances such as people being sick, so we should have enough time to make it up.
Technologies on which we depend might turn out to be unreliable	High	Low	We have made sure that we are aware of the technologies to a good extent, and even still if something were to go wrong, we have accounted for it in our time scheduling.