ZAPP

SMART ENERGY ASSISTANT

FLORIAN DEJONCKHEERE, OTTO HELDT, JONI RAJAMÄKI

THE TEAM

FLORIAN DEJONCKHEERE

OTTO HELDT

JONI RAJAMÄKI

AGENDA

Overview of the solution

Technologies used

Methodology of development

Features of the solution

Roadmap for the final version

Schedule for development

Q&A

ZAPP

Smart energy assistant

- Monitor and control residential energy **infrastructure**
- Predict energy **production** and **consumption**
- Optimize power generation through **curtailment**

Mobile-only (for now)

TECHNOLOGY

Server: Python

- **Django** server framework
- **Scikit-Learn** machine learning

Frontend: TypeScript

- **React** framework
- **TailwindCSS** styling

Deployment: Docker containerization

Github for source code

scrum board

issue tracker

continuous integration

continuous deployment

METHODOLOGY

Kanban framework

Break down work into **small tasks**

Tasks: planned → in progress → done

Pull requests: **code review**

DATA ANALYSIS

Input

- **Dummy** dataset
- **Historical data** from TUAS API

Sanitization

Normalization

Cross-validation

Features

Temperature

Cloud cover

Direct/diffuse radiation

Normal Irradiance

ANALYSIS → PREDICTION → SCHEDULING

DATA ANALYSIS

Train **machine learning model for PV power generation prediction**

Random Forest

At a range of 0 - 1700Wh, mean absolute error of 119 Watts

R^2 score for data: = 0.79

Train **machine learning model for predicting load**

Random Forest

R^2 score for data: = 0.1

PREDICTION

Using the **trained model** and **weather predictions**

Predict energy **consumption** and **production**

Predicted **spot price** from API

ANALYSIS → PREDICTION → SCHEDULING

SCHEDULING

Scheduling

- **Rule-based** constraints: time, price, power source
- Smart **energy** schedule
- Determine best time to enable devices
- Enables **curtailment**
- Enables **peak shifting**

ANALYSIS
PREDICTION
SCHEDULING

DEMO

HTTPS://ZAPP.DEJONCKHEE.RE

COMPLETED

Authentication and **security**: sign in, sign up

Quick overview

Infrastructure configuration

Prediction model and **scheduling** algorithm

ROADMAP

Responsive design

Real-time monitoring (TUAS API)

Infrastructure management

Expand **rule-based** expression system

Integration with **remote energy infrastructure**

TUAS EMS API

Project written in **Python**

Use of existing **API client**

Scheduler: control infrastructure based on **generated schedule**

DEVELOPMENT

Depending on **features** agreed upon

- Improvement of **prediction model**: 2 weeks
- **Deployment**: 1 day
- **TUAS EMS** integration: 2 weeks
- Real-time **monitoring**: 1 week
- Responsive design and **accessibility**: 2 weeks

¿QUESTIONS?

KIITOS!