Kandidatuppsats 2021

Lastkontroll på elnätet genom smartare laddning av elbilar

Samuel Jönsson, Fredrik Lindén, Nora Ojensa, Karl Svantesson,
William Truvé & Christian Weckner

CHALMERS

Institutionen för elektroteknik
Avdelningen för elkraftteknik
Chalmers Tekniska Högskola
Göteborg, Sverige 2021

Lastkontroll på elnätet genom smartare laddning av elbilar

Handledare: Ali Fotouhi , Institutionen för elektroteknik
Handledare: David Steen , Institutionen för elektroteknik
Examinator: Jimmy Ehnberg , Institutionen för elektroteknik

Kandidatrapport 2021
Institutionen för elektroteknik
Avdelningen för elkraftteknik
Chalmers Tekniska Högskola
SE-412 96 Göteborg

Typeset in LaTeX
Göteborg, Sverige 2021

Lastkontroll på elnätet genom smartare laddning av elbilar

Institutionen för elektroteknik
Chalmers Tekniska Högskola

Abstract

Due to the continuing developement of electric vehicles and plug-in hybrids, the overall load on the electric grid is increasing. The specific problem occurs when a large amount of power needs to be distributed during a specific time, such as in the morning, when the industries start up, or around 5 pm, when EV-owners have arrived home from work, and need to plug in. To handle this surge, the charging of EVs can be distributed over a large timeframe, in accordance to the needs of the owner. In this project, a Python based program was developed with the intent of optimizing when, and with what current, an EV will be charged. Furthermore, a graphical user interface was designed, where the user could provide information such as departure time, the current level of charge, among other things.

Sammanfattning

På grund av den ständiga utvecklingen av elektriska fordon som bilar och plug-in hybrider i dagens samhälle belastas elnätet mer än vad det klarar av vid specifika tider, det är ett problem som behöver lösas. Ett sätt att behandla detta problem är smartladdning, vilket är en typ av laddning då fordonet laddas efter behov, belastning på elnätet och pris. Den typ av lösning som tagits fram till är en mikrokontroller med en skärm där användaren skriver in vilka tider den behöver fordonet och en programvara som skapats kontrollerar vilka tider som laddningen kan ske för att minska belastning och pris.

Författarnas tack

- Ali Fotouhi
- David Steen
- Taz Lodder
- Daniel Weerasooriya
- Staffan Truvé
- Sebastian Haglund El Gaidi, Rebase Energy
- Greta Gascon Rudin, Chalmers Studentkårs Kårledning

Ordlista/Förkortningar

Backend:
Backend syftar på funktionaliteten bakom kulisserna som användaren inte kan se. I detta projekt innebär det de beräkningar som ska användas för att optimera strömstyrkan samt funktionerna för att kommunicera med laddaren.

Frontend:
Frontend är den del av ett system som användaren interagerar med. I detta projekt är frontend:en det grafiska gränssnittet som utvecklats.

Flask:
Flask är ett ramverk utvecklat i programspråket Python som används för att underlätta webbutveckling. Det bidrar med grundläggande funktioner som en testserver och har stöd för att lägga till fler funktioner som inte ingår i grundramverket.

REST-API:
REST-API står för Representational State Transfer - Application Programming Interface. Detta fungerar som ett gränssnitt mellan en klient och en server för kommunikation med hjälp av HTTP-standarden.

Flask-REST-API:
Ett tillägg till Flask som underlättar utvecklingen av REST-API:er.

MongoDB:
En databashanterare för ostrukturerad data. I detta projektet användes deras gratis molntjänst.

State of charge eller SoC:
Beteckning som talar om batteriets status på ett elektriskt fordon.

HTTP:
Hypertext Transfer Protokoll, det vanligaste protokollet för datakommunikation över Internet.

Innehåll

Figurer **xiii**

Tabeller **xv**

1 Introduktion **1**
1.1 Syfte . . . 2
1.2 Avgränsningar . . . 2

2 Bakgrund **5**
2.1 Mängd koldioxid i elnätet . . . 6
2.2 Variation av elpris . . . 7
2.3 Smart laddning . . . 8

3 Metod **9**
3.1 Hårdvara . . . 9
3.2 Chassi . . . 9
3.3 Användargränssnitt . . . 10
3.4 Optimeringsmodell . . . 11
3.4.1 Belastningsprognos . . . 13
3.4.2 Solkraftsprognos . . . 13
3.5 Backend . . . 13
3.5.1 Uppbyggnad av laddningsprocess . . . 14
3.5.2 Primär Raspberry Pi . . . 14
3.5.3 Sekundär Raspberry Pi . . . 15
3.5.4 Datorsimulering . . . 15

4 Resultat **17**
4.1 Kommunikationsväg . . . 18
4.2 Optimeringsmodell . . . 18
4.3 Prognoser . . . 20
4.4 Backend . . . 21
4.5 Användargränssnitt . . . 22
4.6 Tester . . . 23
4.6.1 Test 1 8/4-21 . . . 23
4.6.2 Test 2 12/4-21 . . . 24
4.6.3 Test 3 20/4-21 . . . 24
4.6.4 Test 4 23/4-21 . . . 24

4.6.5 Test 5 28/4-21 24
4.6.5.1 Kontrollberäkningar 25

5 Diskussion **27**
5.1 Vad som användes 27
5.2 Alternativ lösning 27
5.2.1 Användargränssnitt och pekskärm 27
5.2.2 Chassi 28
5.3 Optimeringsmodell 28
5.4 Prognoser 29
5.5 Testresultat 29
5.6 Ett samhällsperspektiv 30
5.7 Vid uppföljning av projekt 31
5.7.1 Användarkonton 31
5.7.2 Integration av serverdel 31
5.7.3 Livscykelanalys 32

6 Slutsats **33**

Referenser **35**

Appendices **37**

Figurer

1.1 Antalet laddbara bilar i Sverige år 2012-2020. Från [1] 1
1.2 Nederbörd i Göteborg år 2020 [5] 3

2.1 Export och import av el under diverse tidsintervall 2020/2021. Det första gulmarkerade fältet är ett dygn då Sverige hade nettoimport och det andra gulmarkerade fältet visar tiden då efterfrågan var som högst det dygnet. Från [8]. 6
2.2 Elprisvariation över en vecka. Från [12]. 7
2.3 Belastning kontra vind- och solkraft. Från [13]. Återgiven med tillstånd. 7

3.1 Chassit i sprängvy 10
3.2 Händelseförloppet backend:en byggde på. Systemet skickar en begäran till CTEK:s portal och får tillbaka ett svar. Det gäller för alla händelser förutom de två fetstilade händelserna (NotifyStart och NotifyStop), för dem gäller det omvända. 14

4.1 Den färdiga produkten. Ytterligare bilder finns bifogat i bilaga .10 . . 17
4.2 Kommunikationsväg 18
4.3 Grafer för kostnad och total laddning i kWh 19
4.4 Priser över tid, 13 maj 22:00 - 14 maj 17:00 20
4.5 Belastningsprognos över tid. 20
4.6 Solkraftsprognos över tid. 21
4.7 Två exempel på skärmar i användargränssnittet 22

5.1 Personbilar i Sverige år 2020 uppdelat per drivmedel från trafikanalys. [20] 30

.1 Svar från användartest . . . . . a
.2 Svar från användartest . . . . . b
.3 Svar från användartest c
.4 Manus användartest för användargränssnittet . . . . . d
.5 Huvuddel av chassit . . . . . e
.6 Mellanlägg till chassit . . . . . f
.7 Lock till chassit . . . . . f
.8 Hela chassit . . . . . g
.9 Den färdiga produkten . . . . . h
.10 Produkten sedd från höger, med luftintag synligt i

.11 Produkten sedd från vänster, med utblås synligt . . . . . j
.12 Inmatning för lösenkod . . . . . k
.13 Fel lösenord l
.14 Inmatning för nuvarnade batteriprocent l
.15 Informationsruta för nuvarande batteriprocent . . . . . m
.16 Inmatning för önskad batteriprocent . . . . . m
.17 Informationsruta för önskad batteriprocent . . . . . n
.18 Välj datum och tid . . . . . n
.19 Informationsruta om hur datum och tid väljs, samt tidsbegränssningar o
.20 Datumväljarverktyget . . . . . o
.21 Tidsväljarverktyget . . . . . p
.22 Hur det ser ut när datum och tid är vald . . . . . p
.23 Bilmärkesmeny . . . . . q
.24 Modellmenyn efter att bilmärke har valts . . . . . q
.25 Informationsruta om bilmärkesmenyn . . . . . r
.26 ”Other”-menyn, där självinmatning av batterikapacitet och maximal strömstyrka sker . . . . . r
.27 Informationsruta om batterikapacitet och maximal ström . . . . . s
.28 Välj laddaruttag . . . . . s
.29 Informationsruta om laddaruttag . . . . . t
.30 Sammanfattningsruta . . . . . t
.31 Programmet förbereds för att ta emot nästa användare . . . . . u
.32 Första test av 9 A . . . . . w
.33 Första test av 12 A x
.34 Ström vid uppstart av laddaren x
.35 Test med 6 A strömstyrka i en minut . . . . . y
.36 Test med 12 A strömstyrka i en minut . . . . . y
.37 Test med 9 A strömstyrka i en minut . . . . . z

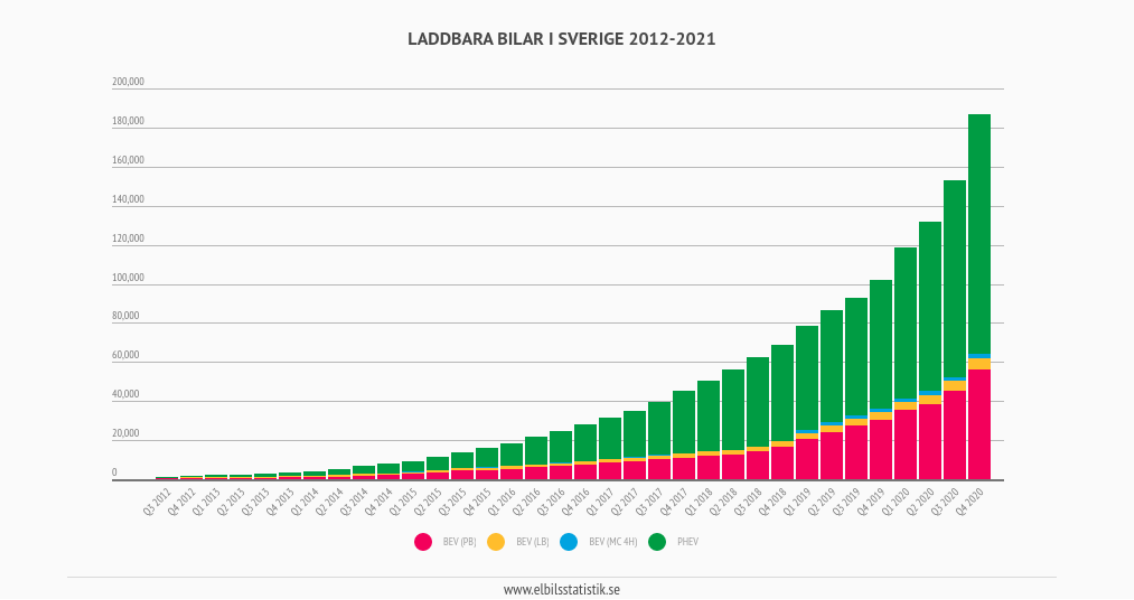
Tabeller

3.1 Parametrar 11

4.1 Strömmätning 25
4.2 Beräknade strömmar jämfört med mätningar 25

1
Introduktion

Antalet elbilar och andra elfordon ökar, både här i Sverige men även runt om i världen. Marknaden rör sig mot ett minskat beroende av fossila bränslen och idag är fyra procent av Sveriges personbilar eldrivna fordon. Denna tydligt uppåtgående trend av antal elbilar kan betraktas i Figur 1.1. Figuren visar att över 180 000 elbilar var i bruk 2020, jämfört med år 2012 då det endast var totalt runt 1000 privata elbilar registrerade i Sverige.



Figur 1.1: Antalet laddbara bilar i Sverige år 2012-2020. Från [1]

Sveriges elnät kommer att utsättas för en högre belastning med den elektrifiering som sker inom transportsektorn, mer specifikt på grund av den laddning som måste göras av elbilarna. Det är beräknat att till 2030 kommer 2,5 miljoner av de 5 miljoner bilar som är registrerade i Sverige idag vara utbytta till elbilar. [1]. Vid ett sådant utfall kan det uppstå problem, där det utmärker sig mest vid laddning av elbilarna vid samma tidpunkter, exempelvis under natten då många laddar sina elbilar. I ett kvarter där 42 procent av alla bilägare har elbil och de alla laddar sina bilar över natten finns det risk att Sveriges elnät skulle bli överbelastat vid kalla perioder under vintern. Sveriges elnät är i dagsläget överdimensionerat och kan hantera högt tryck, men i framtiden löper vi risk att överskrida vår elnätskapacitet. [2]

Både överbelastning och effektbrist kan leda till att strömavbrott kommer att utlö-

sas runt om i landet, vilket kan ha omfattande konsekvenser. Förutom de vardagliga problem det här innebär, kan det ha mycket negativa konsekvenser på känsliga sektorer såsom hemsjukvården. Även korta elavbrott bidrar till stora samhällskostnader [3]. Ett alternativ för att möta elefterfrågan i framtiden är att förstärka elnätet, men det är en kostsam lösning som kommer att medföra stora utgifter för Sverige. En annan mer tekniskt inriktad lösning kommer att utforskas i den här rapporten, nämligen smart laddning. Begreppet redogör för hur man redan nu kan minska den höga belastningen vid kritiska tider genom att styra laddningen av elbilar till tider då elnätet är mindre belastat.

1.1 Syfte

Syftet med projektet är att med hjälp av en mikrokontroller och egen mjukvara optimera och styra laddningen av elbilar som ansluts till en laddstation. Optimeringen kommer ske med hänsyn till användarens egna parametrar, samt utomstående faktorer i form av elnätsbelastning och solkraftsproduktion. För att användaren ska kunna förse optimeringsmodellen med information kommer ett pekskärmsbaserat gränssnitt tas fram. Därmed delas projektet in i fyra delar. Utvecklandet av ett användargränssnitt, ett chassi för mikrokontrollern som all mjukvara kommer att hanteras på, ett inre system som sköter kommunikation mellan mikrokontrollern och laddstationen och optimeringsmodellen där beräkning av laddningseffekten sker.

1.2 Avgränsningar

Projektets fokus ligger på att utveckla en produkt för elbilars laddningstationer som kommer att lägga mindre belastning på Sveriges elnät än att ladda konventionellt och visa ekonomiska fördelar för den enskilde användaren av en sådan produkt. Arbetet utförs som en utveckling av projektet *"Innovative energy management system for smart buildings and grid interactions"* och sker i samarbete med HSB Living Lab [4]. En optimeringsalgoritm kommer att utvecklas där de främsta parametrar är elnätets belastning, elpriser och soldata från HSB Living Lab:s egna solpaneler. Däremot innehas inte tillgång till mätdata i realtid för belastning och dessa kommer därför att beräknas med maskininlärning från historisk data. Produkten som tas fram skall vara klar för användning och kunna placeras på HSB Living Lab:s egen laddningsstation för elbilar, därmed utvecklas också ett grafiskt gränssnitt och en fysisk kontroller med ett tillhörande chassi. Följande punktlista utesluts ur projektet:

- Prisprognos kommer ej genomföras på grund av de höga priserna på API-nycklarna.
  - Istället används historisk data i ett demonstrationssyfte.
- Prognoserna kommer endast se 24 timmar framåt eftersom det är svårare att kalibrera längre prognoser för att få bra resultat.

- Optimeringsmodellen tar inte hänsyn till realtidsdata för belastning på elnätet eller soldata i och med att ingen API-nyckel tillhandagavs eller hittades.
  - Istället undersöks historiska värden som tillhandagavs av HSB Living Lab och bearbetades via Rebase Energy:s plattform.
- Implementation av användarkonton kommer ej att genomföras då det tar mycket serverutrymme och inte är inom projektets syfte. Detta ses som ett utvecklingområde vid vidareutveckling av projektet.
  - En generell lösenkod på programmet implementeras istället för att individuellt identifiera godkända användare.
- Projektet undersöks endast med CTEK:s laddare som är den enda typen av laddare som finns till förfogande.

Projektet tar endast hänsyn till elbilar och är därmed inte en allmän lösning för överbelstning av elnätet. Samma princip kan dock användas för övrig elektronik i våra hem. Det kommer också att antas vara liknade väderförhållanden i Göteborg som det föregångna året 2020, vilket visas i figur 1.2, dvs. en majoritet av dagar med nederbörd.

2020 **Visar år 2020, totalt 1006 mm** Årsöversikt

	Jan	Feb	Mar	Apr	Maj	Jun	Jul	Aug	Sep	Okt	Nov	Dec
1		8,1	4,5		5,7		0,8				13,6	
2	1,1			0,7	4,8			0,7			11,0	
3	2,0	10,5	5,1			0,5	15,2		3,1		5,6	
4		0,3	0,6		0,3	2,9	7,7	1,1	0,8	17,6		
5	2,7					7,2	16,5	9,4	7,5	0,8	0,2	
6	1,0			0,1		8,3	8,0	0,3	0,3	2,3		
7	3,4	0,3					1,3		4,0	2,0		0,2
8		6,1	2,4	0,8					3,8	13,6		2,6
9	5,3	23,1	1,4		0,4				1,2	9,2	0,2	2,5
10	0,9	0,3	13,3		4,4					8,2		0,2
11	8,5	2,6	13,1	13,0					6,8	3,8	0,9	
12	0,1		11,7		0,3				9,2	0,2	0,6	1,6
13	1,7		1,3				6,1		11,0		3,5	0,2
14	13,9	4,1		0,9			12,5				0,2	4,5
15	1,3	19,2	8,6								3,5	4,8
16		14,6	0,1			0,5				0,2	3,1	
17	3,5	3,1	5,0							0,2	12,4	15,2
18	3,9	1,4	0,5		1,4		0,7				7,0	11,6
19		0,3			0,3	4,9	3,1	2,6			1,5	1,6
20		13,4				17,9		1,2		13,4	5,4	
21		4,8				0,5		4,0		30,7	13,5	2,6
22		3,6			3,8	1,7		3,4		6,0	2,0	5,2
23	0,7	0,6			9,8		7,0		0,9	5,4	5,9	0,2
24		2,7			2,2		12,9	0,2	2,4	0,5	4,0	0,2
25		9,8					0,2	4,0	24,4	7,0	2,8	
26	0,2			0,1		1,0	1,5	0,4	0,1	0,4		26,0
27	9,6	0,1		9,0		6,5	6,9			8,6		24,6
28	10,5			6,9		2,5	10,0	0,6	0,2	8,0		1,9
29		8,4				5,5	1,7	0,1		1,3		3,0
30	14,6	×		6,9		3,3				1,6	6,6	1,9
31	3,0	×		×		×			×	3,2	×	4,0
Σ	88	137	68	38	33	63	112	28	76	144	104	115

Figur 1.2: Nederbörd i Göteborg år 2020 [5]

2

Bakgrund

Det problem som uppstår när en större andel av fordonen drivs av el är att belastningen på elnätet ökar. Annan elförfrågan ökar också i samhället med bland annat tillväxten av techföretag och utökade industrier, men i det här projektet ligger fokus på elbilar och deras påverkan. Kraftnäten i de flesta länder är inte utformade för att klara av den nätbelastning som ett ökat antal elbilar kommer medföra. Att endast utöka elnätet är inte heller socioekonomiskt hållbart. Boston Consulting Group uppskattar att det kommer kosta mellan $1700-$5800 per fordon för att uppgradera elnätet i USA fram till 2030 [6].

Problemet handlar främst om att topparna av belastningen kommer bli större då många laddar sina elbilar samtidigt, exempelvis när de kommer hem från jobbet. Om vi kan omdirigera viss elanvändning vid hög belastning på elnätet till tider då elnätet upplever mindre efterfrågan, kan fler människor ladda elbilen utan att elnätet behöver förstärkas.

Det är inte bara antalet eldrivna enheter som förändras, utan även hur elen produceras. Sverige har som mål att vid 2040 ha en 100 % förnybar energiproduktion, vilket innebär en minskad användning av kärnkraft. Kärnkraftverken har en jämn och stabil elproduktion, till skillnad från den vindkraft som delvis kommer fylla utrymmet. Elproduktionen från vindkraftverk är helt beroende av att det blåser, vilket leder till att tillgången och kostnaden kan variera under dygnets alla timmar. Detta öppnar upp för risken att det är helt vindstilla under en tung belastningsperiod, och vi behöver då köpa in utländsk el. [7].

Figur 2.1 visar import och export av el till och från Sverige och närliggande länder under olika tidsintervall. Det gulmarkerade fältet visar den första januari 2021 då Sverige nettoimporterade 15 124 GWh el, emellertid är det värt att notera att det varierar mycket på efterfrågan under dygnet. Till exempel kan det betraktas i figuren att det importerades 281 GWh el klockan 15-16 och 1205 GWh el klockan 17-18. Förutom den ekonomiska nackdel som importering av el har för konsumenterna, är det också speciellt ofördelaktigt ur ett ekologiskt perspektiv när el måste importeras från Tyskland och Polen som framställer stora delar av sin el av fossila bränslen.[8]

	SE - NO	SE - FI	SE - DK	SE - DE	SE - PL	SE4 - LT	Netto
Helår							
2020	7 859 654	-18 475 698	-3 772 862	-2 116 890	-3 818 869	-4 651 459	-24 976 124
Månad							
21 - Jan	962 992	-1 237 411	-189 150	-172 732	-255 489	-298 640	-1 190 430
20 - Dec	1 079 677	-1 525 165	-353 385	-236 255	-345 339	-406 044	-1 786 511
20 - Nov	240 424	-1 398 703	-92 709	-100 166	-328 061	-368 474	-2 047 689
Vecka							
04 - 21	283 384	-235 719	6 818	6 000	-52 445	-49 782	-41 744
03 - 21	130 095	-278 991	1 451	-35 237	-85 788	-98 148	-366 618
02 - 21	220 959	-324 261	9 572	-24 494	-63 696	-95 037	-276 957
01 - 21	224 660	-267 117	-111 162	-78 162	-33 650	-28 845	-294 276
Dygn							
01-02-2021	36 816	-45 152	15 448	10 242	-3 548	1 318	15 124
31-01-2021	22 973	-34 144	8 673	125	681	-3 666	-5 358
30-01-2021	29 978	-21 381	-2 071	2 317	-11 992	-7 646	-10 795
29-01-2021	57 357	-43 866	20 179	5 449	-9 728	-7 984	21 407
28-01-2021	45 661	-40 192	8 724	6 295	-10 936	-11 341	-1 789
Timmar 01-02-2021							
15 - 16	1 871	-2 561	610	596	-177	-58	281
16 - 17	1 717	-2 512	753	596	-3	144	695
17 - 18	1 962	-2 446	463	596	281	349	1 205
18 - 19	1 960	-2 577	471	596	210	276	936
19 - 20	1 984	-1 941	402	157	-246	158	514

Figur 2.1: Export och import av el under diverse tidsintervall 2020/2021. Det första gulmarkerade fältet är ett dygn då Sverige hade nettoimport och det andra gulmarkerade fältet visar tiden då efterfrågan var som högst det dygnet. Från [8].

Istället för att importera el eller öka elproduktionen för att möta efterfrågan, samt uppgradera hela elnätet för att undvika överbelastning är det snabbare och eventuellt billigare för både privatpersoner och samhället i stort att kontrollera när topparna på elanvändningen sker. Det vill säga att man sprider ut tiderna då elnätet belastas och produktionsefterfrågan sker genom att kontrollera när en elbil laddas. Problemet kan även generaliseras till att justera fler produkters belastning än bara elbilar. Exempelvis att man kontrollerar när en tvättmaskin används för att minska belastningen på elnätet och produktionskapaciteten, samt kostnaden för en privatperson.

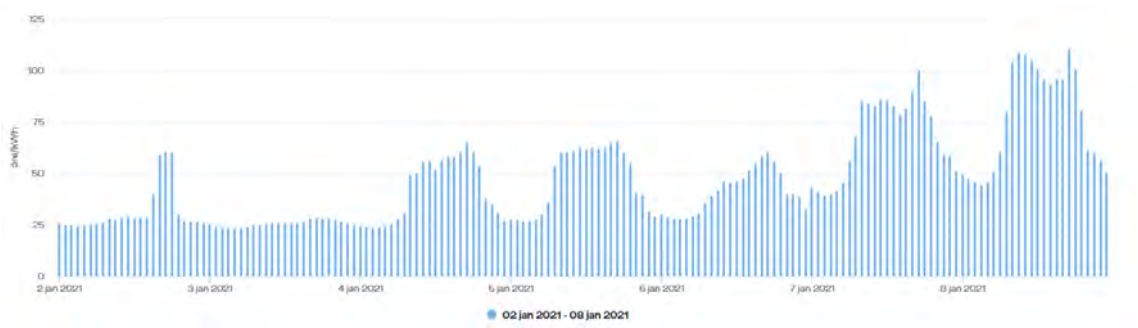
2.1 Mängd koldioxid i elnätet

Den svenska elproduktionen släpper inte ut mycket koldioxid då den främst producerar el av vatten- , kärn- och vindkraft. Det gör att Sverige har ett koldioxidutsläpp på ca 13.0 gCO_2/kWh [9]. Den Nordiska elproduktionen har ett utsläppsmedelvärde på 125.5 gCO_2/kWh under perioden 2005 - 2009, alltså nästan 9 gånger mer än den svenska elproduktionen 2014 [10],[9]. Detta påvisar att det är miljövänligare att använda svenskproducerad el och att det ej är fördelaktigt att importera el sett utifrån ett ekologiskt perspektiv.

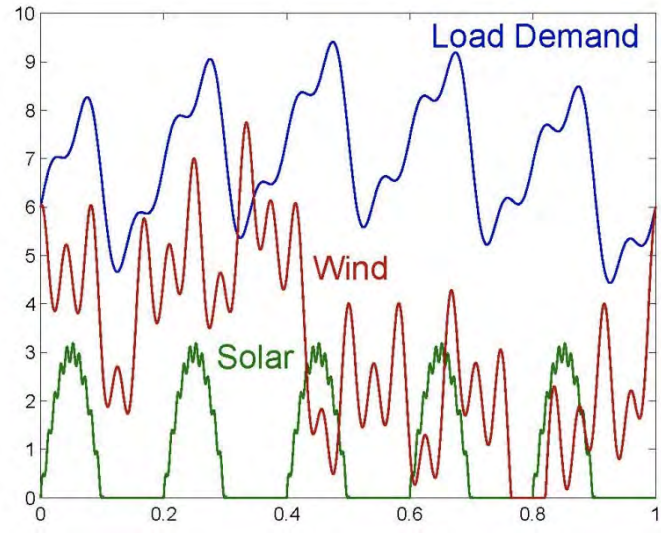
2.2 Variation av elpris

På grund av att Sverige har målsättningen att driva landet på 100 % förnybar energi kommer elpriserna framöver att variera kraftigare än i dagsläget. Anledningen till skillnaden är att förnybar energi oftast inte är planerbar, till skillnad från kärnkraft. Generellt har priset berott på användning, det vill säga högre pris vid högre användning, men med varierad produktion kommer även det behövas tas hänsyn till. Det betyder alltså att det kommer vara högre priser under vissa tider inom en snar framtid, då våra energikällor naturligt producerar mindre el. [11].

I figur 2.2 syns ett tydligt mönster av elprisets variation. Från klockan 10:00 till 20:00 är priset mycket högre än vid andra tider vilket korrelerar med att belastningen är högre. Det skapar problem för framtiden då målet är att behålla samma elnät men använda mer förnybar energi. Det leder till att topparna på priset kommer öka om användarna inte är mer flexibla med sin användning. I figur 2.3 visas förhållandet mellan belastning och förnybar energi, i detta fall vind- och solkraft. Om det exempelvis finns mycket vindkraft i nätet bidrar det till lägre elpris.



Figur 2.2: Elprisvariation över en vecka. Från [12].



Figur 2.3: Belastning kontra vind- och solkraft. Från [13]. Återgiven med tillstånd.

2.3 Smart laddning

I och med att laddningsbara fordon blir allt mer populära uppkommer problem när dessa fordon ska laddas, då elkraften dras från elnätet. För att minska lasten på nätet krävs ny teknik som kan hantera laddningen på ett effektivt sätt så att nätet inte blir överbelastat. I dagens samhälle kontrollerar man inte laddningen i större utsträckning, utan el dras direkt när man kopplar in sin produkt, vilket leder till större belastning på elnätet under specifika timmar då fordonen laddas. Vid användning av smart laddning så minskar både belastningen på nätet och spotpriset blir lägre. Det finns även fördröjd laddning vilket betyder att användarna uppmuntras till att ej ladda under högbelastningsperioder på grund av prisökningen, denna metod fungerar endast då användarna har varierande pris i sitt el-avtal[14]. Enligt [14] minskar belastning på transformatorn med 20 % och belastningen i kablarna med 10 %. För att utnyttja den förnybara energin som i annat fall hade exporterats är det fördelaktigt att ladda då det finns överskott av exempelvis vindkraft kontra belastning, som vid ett tillfälle figur 2.3.

3
Metod

För att minska belastningen på elnätet utvecklades ett program som kan kommunicera med och styra laddstationer för elbilar.

Projektet delades upp i fyra delar:

- Användargränssnitt, där användaren matar in sina önskemål.
- Optimeringsmodell, där beräkningen av laddningseffekten till elbilen sker.
- Backend, där kommunikationen mellan mikrokontrollern och laddstationen sköts.
- Chassi, det yttre höljet som låter oss montera mikrokontrollern på laddstationen.

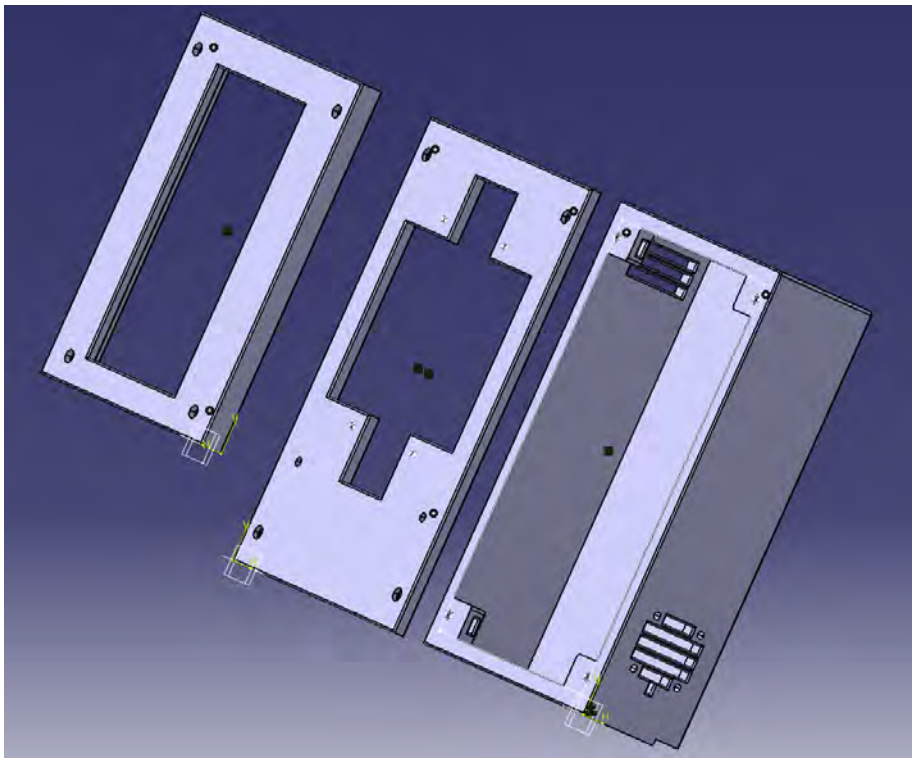
3.1 Hårdvara

Inför valet av hårdvara fanns några krav. Den valda mikrokontrollern skulle ha stöd för Python samt ha möjlighet att ansluta en pekskärm. Med detta i åtanke valdes en Raspberry Pi 4b [15], dels för att den uppfyller dessa krav, men även för att plattformen har haft tid att mogna. För att inte begränsas av hårdvaran valdes modellen med 4 GB internminne, istället för den billigare 2 GB varianten.

Utöver Raspberry Pi:en användes en kapacativ 7” pekskärm från The Raspberry Pi Foundation [16]. Den har en upplösning på 800x480 pixlar, vilket ansågs vara lagom för den funktionalitet som eftersträvades med användargränssnittet.

3.2 Chassi

På grund av väderförhållandena i Sverige var avsikten att ett vattentätt chassi skulle konstrueras för att öka livslängden på produkten, samt göra det enklare att installera den på laddaren. Först togs mått på laddstationen för att göra det möjligt att installera chassit ovanpå. Chassit skapades som en CAD-modell för att sedan 3D-utskrivas i hårdplast. För att Raspberry Pi:en skulle passa i chassit krävdes lite slipning. För att öka kyleffekten till mikrokontrollen installerades en liten fläkt i chassits nedre del och en ventil i det övre hörnet samt en kylfläns på CPU:n. Chassit skruvades ihop med rostfri skruv, mutter samt silikon i fogarna för att öka vattenresistensen.



Figur 3.1: Chassit i sprängvy

3.3 Användargränssnitt

Produktens grafiska gränssnitt utvecklades med Pythonbiblioteken Kivy och Kivy MD, där det sistnämnda bygger på Googles egna designspråk Material Design. Biblioteken är lämpliga då de är framtagna med just pekskärmsapplikationer i åtanke. Det tillhörande Kivy Modeling Lanugage (KV) möjliggjorde ytterligare en separationsnivå mellan designelementen och den egenskrivna logiken.

Målet har för gränssnittet varit att möjliggöra en enkel och behaglig användarupplevelse och besluten som tagits har beaktat detta med grund i standarder för verksamhetsområdet och användartest på produkten. Begränsningar har innefattat genomförbarhet i det givna tidsintervallet och resurser. Ett exempel på detta är valet att det grafiska gränssnittet är uppbyggt som en guide, med inmatning av enskilda parametrar på varje sida istället för ett formulär där alla inmatningsfält är synliga på en och samma sida. Detta för att minska den kognitiva belastning som många inmatningsfält medför för användaren, en teknik som vi idag kan se stora företag använda för sina produkter, till exempel Microsoft med sin installationsguide. [17]

För att säkerhetställa det grafiska gränssnittets funktionalitet och användarvänlighet planerades ett användartest. Testet var beräknat till att utföras på 5 olika individer. Antalet är valt utefter en studie som Jakob Nielsen, en välkänd forskare inom området, publicerat. Studien påvisade att över 75 procent av alla användbarhets-

problem i ett program upptäcks vid test av 5 användare och är rekommenderat som en standard.[18] Testet var utformat som en observation då användarna fick använda produkten och utföra förutbestämda uppgifter så att observatören lätt kunde se vad för problem användaren hade med applikationen. Som ett komplement till detta utfördes en kort semistrukturerad intervju efter observationen, där användaren fick uttrycka sina tankar om applikationen. Fördelen med de här metoderna för att samla data är att resultatet kommer ge kvalitativ data som ger mer insikt än kvantitet data, vid det här antalet testanvändare. Blandningen av att kunna observera vad användaren gör och användarens egna formulerade tankar ger också en djupare insikt. Testets manuskript, med observationsuppgifter och intervjufrågor finns bifogat i bilaga .4

3.4 Optimeringsmodell

Optimeringsmodellen löses som ett ”mixed-integer linear programming” (MILP)-problem för att kunna ta hänsyn till de parametrar som projektet önskar. MILP innefattar de problem där vissa parametrar tillåts vara heltal, medan andra tillåts vara icke heltal. En vanlig metod för att lösa sådana problem är ”Branch and Bound”-metoden som är en slags ”söndra och härska”-metod. I arbetet används Pythonbiblioteket SciPy för att lösa optimeringsproblemet. De parametrar som tagits hänsyn till syns i tabell 3.1.

Tabell 3.1: Parametrar

Parametrar	**Enheter**
Current limit	Ampere
Chargetime	YY:MM:DD:HH:MM
Battery capacity	kWh
Current battery status	Procent
Desired battery status	Procent

Optimeringsmodellen kan beskrivas matematiskt som

$$\sum_{i=1}^{n} Pris_i \frac{X_i * V * \frac{\Delta t}{60}}{1000} =' Totalt\ pris' \tag{3.1}$$

Där ’n’ är laddningstiden i antal 5-minuters intervall och ’X’ är ström i Ampere. Spänningen V är i detta fallet 230 Volt och Δt är 5 minuter eftersom modellen uppdateras var 5:e minut. Genom att dela Δt med 60 samt hela summan med 1000 omvandlas det till hela kr, då priset är i kr/kWh.

Optimeringsmodellens mål är att minimera summan i ekvation 3.1 genom att ladda de tider priset/belastningen är som lägst, alternativt när vi får överskott från sol-

produktionen. Dessutom ska modellen även uppfylla följande villkor nedan, där P i ekvation 3.5 är antalet kWh som krävs för att ladda bilen.
Dessa villkor innebär att laddaren tvingas ladda de tider då solkraftsprognosen ger en tillräcklig effekt. Priset räknas inte bort från elen som kommer från solpanelerna, detta påverkar inte modellen men kan vara bra att känna till eftersom det inte är det faktiska priset som beräknas i ekvation 3.1. Detta scenario kan göra att ekvation 3.5 inte går att lösa, men det är inte heller ett problem, fordonet kommer ändå att ladda endast de timmar som angivits och när laddningen är tillräcklig kommer den att avbrytas. Ett annat exempel då ekvation 3.5 inte går att lösa är då laddningen omöjligt kan genomföras på den tid som angivits, en bil kan till exempel inte laddas fullt på 10 minuter. I detta scenario kommer bilen att laddas med full kapacitet så länge den är inkopplad eller tills tidsangivelsen passerats.

$$Y = Solkraftsprognos - Belastningsprognos \tag{3.2}$$

$$Z = \begin{cases} 0, & \text{Om } Y \leq 0 \\ Strömbegränsning, & \text{Om } Y \geq Strömbegränsning \\ Y, & \text{Övriga fall} \end{cases} \tag{3.3}$$

$$Z \leq X_i \leq Strömbegränsning \tag{3.4}$$

$$\sum_{i=1}^{n} \frac{X_i * V * \frac{\Delta t}{60}}{1000} = P\ kWh \tag{3.5}$$

I de fall ett fordon ska ladda under en period som är längre än 24 timmar framåt ser den över elpriset för nästkommande 24 timmar och väljer laddningstiden som bilen ska ladda utifrån priset, prognoserna, samt användarens input. Som exempel om laddningstiden är 36 timmar kommer modellen först undersöka timmarna 0-24, efter 2 timmar 2-26, och efter 12 timmar 12-36. Alltså uppdateras modellen vid varje anrop vilket är bra då den alltid tvingas ladda när solcellerna producerar en tillräcklig effekt, och den kan i vissa fall undvika att använda sig av elnätet helt och hållet.

I de fall då det är mindre än 24 timmar återstående laddningstid kommer modellen undersöka samma timmar, med undantaget att de timmar som passerat inte längre räknas med.

Sammanfattningsvis fungerar modellen genom att dela upp dessa 24 timmar i fem minuters intervall. Sedan beräknas hur mycket ström som ska tillföras för varje enskilt fem minuters intervall. Detta leder till att bilen laddar då solcellerna ger tillräcklig effekt alternativt den tiden då priset är som lägst. I dagsläget använder sig modellen av genomsnittliga timpriser, detta medför att vi även tar hänsyn till belastning eftersom priset har en stark korrelation gentemot lasten på elnätet.

En av parametrarna som optimeringsmodellen beaktar är en solkraftsprognos. Denna är framtagen med Rebase Energy, en plattform som tränar AI-modeller för prognoser av energiproduktion [19]. Den inmatade datan som försågs av HSB Living

Lab bestod av solkraftsproducerad effekt över tid, vanligtvis med mätningar på timbasis.

3.4.1 Belastningsprognos

För att optimeringsmodellen skulle kunna ta hänsyn till HSB Living Labs belastning, behövdes en progons. HSB hade försett oss med historisk belastningsdata, i form av (Tidsstämpel, Effekt) för varje minut under drygt tre års tid (jan 2017 - mar 2020).
Den första ansatsen var att med hjälp av Pythonbiblioteket TensorFlow ta fram en modell som korrelerar tid och effekt, för att sedan förutspå kommande belastningsvärden. Då gruppen saknade tidigare erfarenhet av maskininlärning och TensorFlow, togs beslutet att istället använda Rebase Energy [19]. Rebase är en plattform där man skapar anläggningar, för exempelvis solkraftsproduktion, vindkraftspoduktion, eller belastning. Den inmatade informationen blir sedan parametrar för den prognosmodell som Rebase använder. Denna anläggning går sedan att träna med historisk belastningsdata, på formatet (Tidsstämpel, Förbrukad Effekt).
För att skapa en belastningsprognos behövdes det således bara matas in koordinater för HSB Living Lab, samt välja vilka väder- och datumparametrar som prognosen ska ta hänsyn till. Därefter matades datan från HSB in. Datan nedsamplades till en datapunkt varje kvart, istället för varje minut, då Rebase hade problem med att hantera mängden.
När modellen var färdigtränad var det bara att använda Rebase:s API för att hämta den färdiga prognosen. Prognosen består av två arrayer, ([Tidsstämpel], [Förväntad Belastning kW]), med upplösning på en datapunkt per timme. Då optimeringsmodellens optimeringshorisont är på de kommande 24 timmarna, plockas endast dessa ut ur prognosen.

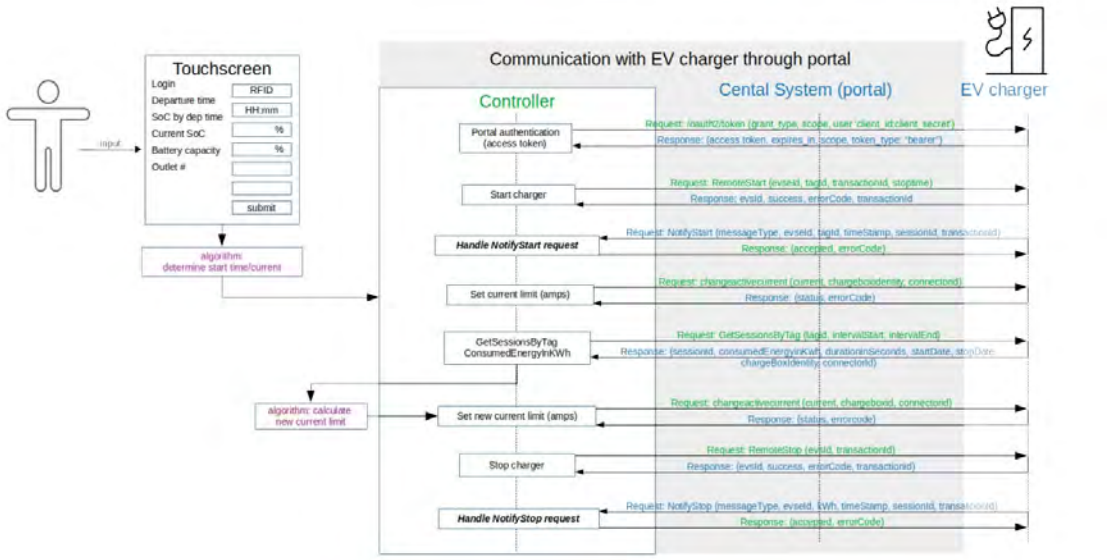
3.4.2 Solkraftsprognos

Istället för att ta fram en egen prognos användes återigen Rebase Energy. För att skapa en solkraftsanläggning matar man in koordinater, vilken rikting solpanelerna sitter i, samt vilken vinkel de är monterade i. Modellen tränas, likt belastningsmodellen, med historisk produktionsdata (Tidsstämpel, Producerad Energi) , återigen försedd av HSB Living Lab.
När modellen är tränad kan man via Rebase:s API hämta prognosen, som består av ([Tidsstämpel], [Förväntad Produktion]) för varje kvart de kommande fem dygnen, inklusive nuvarande dag, med start kl 00:00. Precis som för belastningsprognosen använder optimeringsmodellen de kommande 24 timmarna, vilket innebär att de kommande 96 (24 timmar $\cdot$ 4 kvartar) extraheras från prognosen.

3.5 Backend

Backend:en består av två delar. En lokal primär del som hanterar kommunikationen mellan användargränssnittet och optimeringsmodellen. Den andra är en extern sekundär server som går på en annan Raspberry Pi 4b. På grund av att backend:en är

delad på två separata enheter används MongoDB som databas för att skicka samt manipulera data mellan dem. Backend-systemet som helhet utgår från händelseförloppet i figur 3.2.
Anledningen till varför systemet delades upp på två separata enheter är på grund av att server-delen av backend:en behöver en konstant internetuppkoppling för tester, vilket under utveckling av produkten inte var möjligt med endast den lokala enheten, då den flyttades runt mellan olika utvecklare i projektgruppen. Utöver att internetuppkopplingen gav upphov till problem, upptäcktes det att Raspberry Pi 4b blir väldigt varm och överhettar med enbart backend igång. Därmed, för att förenkla utvecklingsprocessen och inte behöva byta API:ets IP-adress vid varje flytt togs beslutet att använda en extern server.



Figur 3.2: Händelseförloppet backend:en byggde på. Systemet skickar en begäran till CTEK:s portal och får tillbaka ett svar. Det gäller för alla händelser förutom de två fetstilade händelserna (NotifyStart och NotifyStop), för dem gäller det omvända.

3.5.1 Uppbyggnad av laddningsprocess

Som det illustreras i figur 3.2 sköts kommunikationen med hjälp av HTTP-begäran och HTTP-svar. Laddningsprocessen kan därmed ses som tvådelad, den ena är de HTTP-kommandon som används för extern kommunikation och den andra är exekveringsloopen i programmet. Varje HTTP-kommando är uppbyggt på ett liknande vis, endast den specifika informationen, som exempelvis datan, skiljer sig. Exekveringsloopen i sin tur strukturerar upp när varje kommando ska skickas och med vilken information. Den sköter även den interna kommunikationen mellan backend och optimeringsmodell.

3.5.2 Primär Raspberry Pi

Den primära Raspberry Pi:en kör kontinuerligt användargränssnittet på pekskärmen. När användaren försett all information som är nödvändig för en laddningsses-

sion, skapas ett backend-objekt. Användargränsnittet återställs, och är redo för att ta emot parametrar för nästa bil. Backend-objektet är kopplat till det valda uttaget på laddaren, vilket gör att två elbilar kan laddas samtidigt, oberoende av varandra. När ett backend-objekt har skapats startar det själva laddningsprocessen.

Laddningsprocessen sköter kommunikationen med laddstationens API för att starta laddaren, ändra strömförsörjningen, stänga av laddaren, etc. För att få den korrekta datan att skicka till laddstationens API kallar laddningsprocessen på optimeringsmodellen.

Under laddning kallar laddningsprocessen på optimeringsmodellen och uppdaterar strömstyrkan var femte minut. Internt uppdateras bilens nuvarande procentuella laddning samtidigt. Då informationen inte går att få från laddningsstationen görs en uppskattning av bilens procentuella laddning avrundat till närmaste heltal.

Laddningsprocessen stoppas då den uppskattade procentuella nivån har nått den inställda nivån användaren ville uppnå. Den stoppas även om tiden vid uppdatering överskrider den inställda avfärdstiden eller om bilen inte längre är inkopplad. När laddningsprocessen har stoppats ser backend-objektet om det andra uttaget används. Ifall det inte används stängs laddstationen av, annars fortsätter den vara igång.

3.5.3 Sekundär Raspberry Pi

Den sekundära Raspberry Pi:n kör ett Flask-REST-API. Detta är en väldigt enkel server som tar emot och skickar tillbaka JSON-objekt, samt uppdaterar en databas. Som tidigare nämnt används MongoDB som databashanterare. När laddningsprocessen skickar ett meddelande till laddstationens API att laddstationen ska startas eller stoppas skickar API:et en begäran tillbaka som behöver besvaras för att fortsätta. Den adress som API:et skickar till är hårdkodad, och måste förses manuellt via mejl till ansvarig på CTEK.

Begäran innehåller informationen som laddstationen tog emot som då kan kontrolleras att den stämmer. Datan lagras på en MongoDB-databas som den primära Raspberry Pi:en kan kontrollera mot att datan är korrekt. Servern skickar sedan ett svar tillbaka att begäran togs emot och att det är okej att starta respektive stänga av laddstationen.

3.5.4 Datorsimulering

Innan tester med en bil utfördes säkerställdes det att produktens olika delar fungerade tillsammans och att produkten kunde kommunicera med CTEK:s server. Som nämnt tidigare i 3.5.1 så är laddningsprocessen tvådelad. Detta gjorde att simulering av varje HTTP-kommando kunde ske separat. För att snabbt testa på ett robust vis om strukturen på ett kommando var korrekt användes Postman. Med Postman kan man skicka HTTP-begäran för att testa ett API och se att ens meddelanden

kommer fram till servern. Sen säkerställdes det att de programmerade kommandon fungerade. Detta gjordes genom att köra en Python-fil med varje kommando och returnera de HTTP-statuskoder som fås tillbaka.

Nästa steg i datorsimuleringen var att testa exekveringsloopen. Då exekveringsloopen ska vänta i fem minuter mellan varje uppdatering simulerades även tiden genom att lägga till fem minuter mellan varje iteration i loopen. Till skillnad från att använda datorns systemklocka vilket görs i slutprodukten.

När systemet som helhet skulle testas gjordes det först endast med backend:en och optimeringsmodellen. Ett testscript konstruerades för att förse backend:en med hårdkodade parametrar. På så vis kunde de delarna av programmet kontrolleras utan att stega igenom användargränssnittet. När backend:en och optimeringsmodellen gav upprepat godkända resultat startades användargränssnittet upp för en slutgiltig kontroll av hela systemet.

4

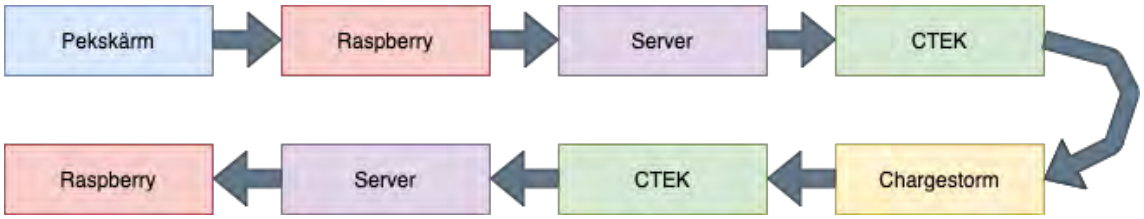
Resultat

Slutrodukten är uppbyggd av flera delar, vilka förklaras i nedanstående sektioner. Den fungerar genom en optimeringsmodell som tar hänsyn till belastnings- och solkraftsprognoser och desstom implementerar användarinmatning för att beräkna de tidsintervall som elnätet är minst belastat.

Figur 4.1: Den färdiga produkten. Ytterligare bilder finns bifogat i bilaga .10

4.1 Kommunikationsväg

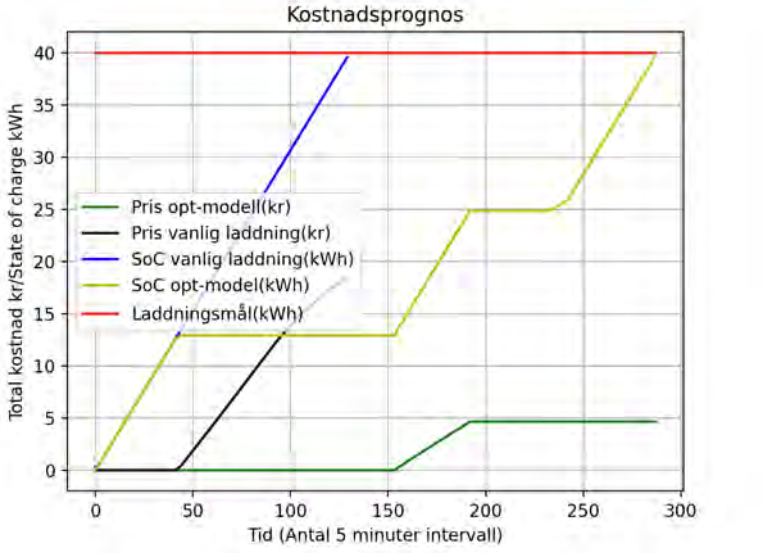
Prototypen fungerar så att en användare använder pekskärmen för att mata in sina inställningar för laddningskapacitet och diverse andra inställningar som tydliggjorts i tidigare kapitel. Pekskärmen är ihopkopplad med Raspberry Pi:en som kommunicerar med servern. När programmet körs skickar Raspberry Pi:en en acknowledgement till CTEK som i sin tur kommunicerar internt med Chargestormladdaren. Laddaren skickar i sin tur ett OK till CTEK som kollar med servern och som skickar ett OK till Raspberry Pi:en, sedan startar laddaren. Problemet som uppstod under testerna var i sista steget från CTEK till servern där de inte hade den korrekta konfigurationen för servern. Misstanken var att CTEK inte ändrade sin serverkonfiguration efter att begäran samt skickade information om hur de skulle utföra denna, då den tidigare servern fungerade utan några problem. Detta sammanfattas i figur 4.2



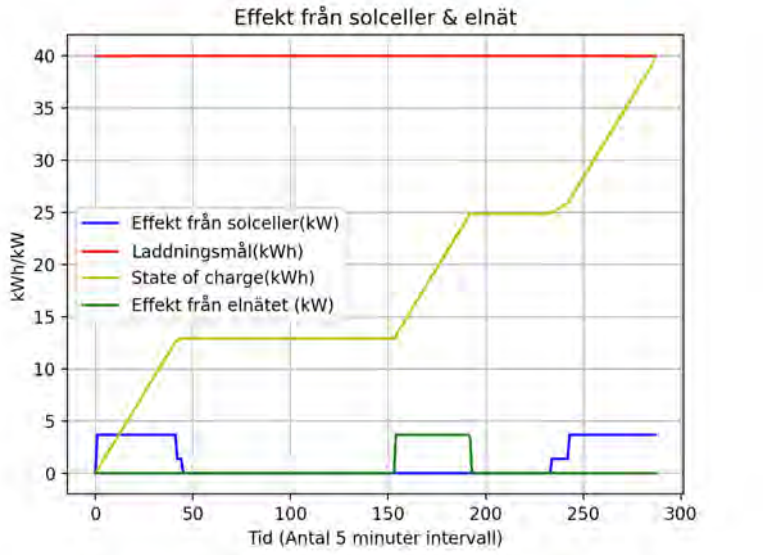
Figur 4.2: Kommunikationsväg

4.2 Optimeringsmodell

Resultatet av optimeringsmodellen är att laddstationen maximerar användandet av effekten som kan fås ut från solpanelerna. När effekten från solpanelerna är låg kontrollerar modellen istället när elpriset är som lägst och väljer att ladda dessa tider. I figur 4.3 demonstreras hur optimeringsmodellens prognos kan se ut, samt hur priset kan skilja sig mot en laddning utan optimering. I graferna nedan har kostnaden av effekten från solcellerna antagits vara 0 kr.

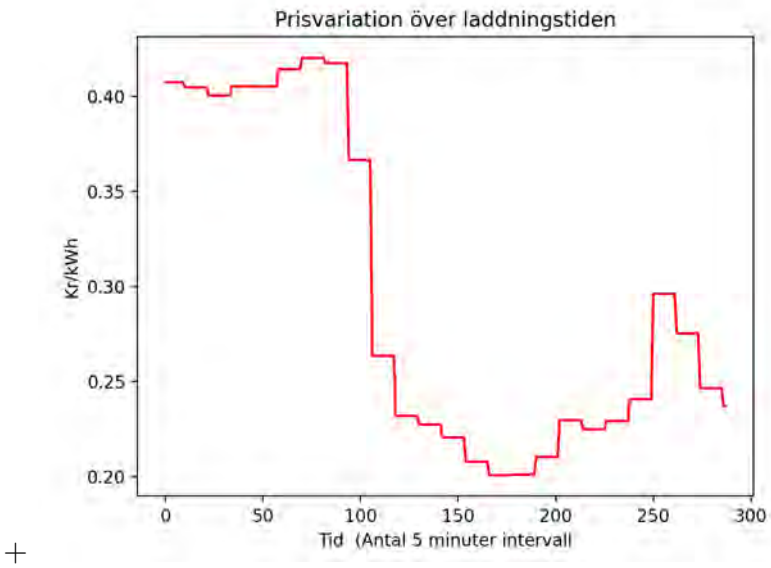


(a) Kostnad för olika laddningsalternativ



(b) Laddningsprognos

Figur 4.3: Grafer för kostnad och total laddning i kWh



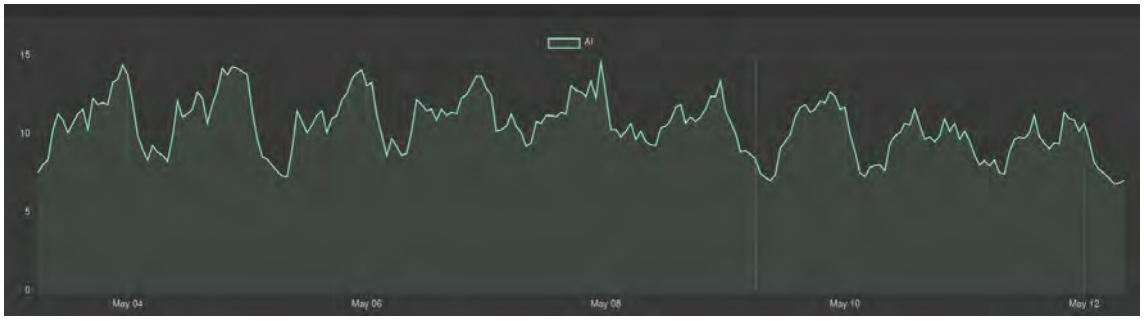
Figur 4.4: Priser över tid, 13 maj 22:00 - 14 maj 17:00

Utöver det som kan läsas från figuren så uppdateras optimeringsmodellen vid varje anrop från backend. Eftersom modellen som längst blickar 24 timmar framåt kan prognosen ändras och eventuellt, beroende på hur länge bilen ska ladda, kan optimeringsmodellen detektera att den kan få ut mer effekt från solpanelerna.

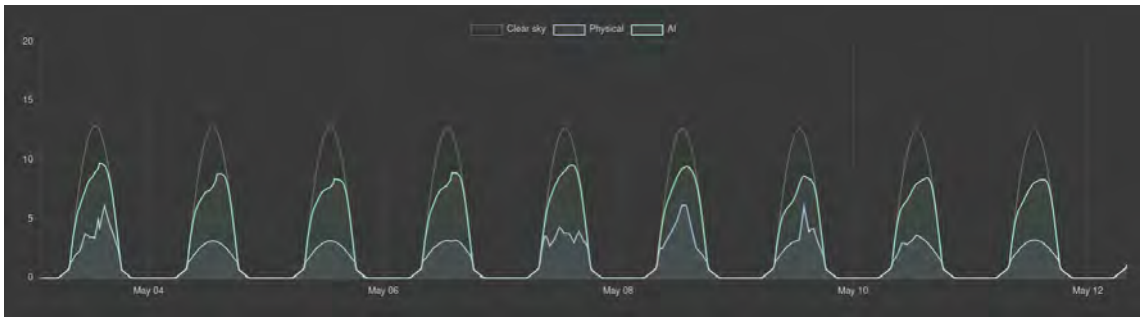
Det slutgiltiga resultatet är en modell som väljer att ladda när solpanelerna kan producera en tillräckligt hög effekt, alternativt på natten då belastningen och priset är lägre än mitt på dagen.

4.3 Prognoser

I slutprodukten användes två olika prognoser; en för belastning och en för solkraftsproduktion. De båda prognoserna är framtagna med Rebase Energy:s prognosmodeller, som tränats med historisk data från HSB Living Lab. Det färdiga resultatet förutspår ett effekttal vid en viss tid på dygnet. Prognoserna finns dels visualiserade på Rebase:s hemsida, men även tillgängliga som JSON-objekt via deras API. I figur 4.6 är det den gröna kurvan som använts i optimeringsmodellen.



Figur 4.5: Belastningsprognos över tid.



Figur 4.6: Solkraftsprognos över tid.

4.4 Backend

Vid datorsimuleringar av backend:en upptäcktes problem i logiken som snabbt kunde lösas. Ett av problemen var att datan som togs emot från optimeringsmodellen av backend:en var i fel format. Optimeringsmodellen returnerar en lista med alla beräknade värden. Funktionen som skickar strömstyrkan till laddaren tar endast emot ett värde. Lösningen var därför att plocka ut det första värdet ur listan och skicka det.

Ett problem som är en fortsättning på det är att i det fallet att laddningen överskrider den avfärdstid som användaren skrivit in blir värdena oanvändbara. Detta då optimeringsmodellen beräknar med avfärdstiden som mål. Överskrider laddningen den tiden finns det inget mål för optimeringsmodellen att beräkna mot. Lösningen på det problemet var att stoppa laddningen om antingen bilen har nått den procentuella laddning användaren ville ha, eller vid det fallet att tiden vid laddning överskrider den satta avfärdstiden. Även i det fallet då bilen inte uppnått den laddning användaren ville nå.

En liten bugg som fick en stor konsekvens var beräkningen av nästa tidspunkt då exekveringsloopen ska skicka nästa strömstyrkevärde. Problemet var att variablerna som användes för att jämföra den nuvarande tidspunkten och nästa tidspunkt uppdaterades samtidigt vilket ledde till att programmet aldrig gick vidare i exekveringen. Med andra ord fastnade programmet i ett stadie den aldrig kunde komma ut ur. Lösningen var att endast uppdatera variabeln med nästa tidspunkt då programmet skickade nästa strömstyrkevärde.

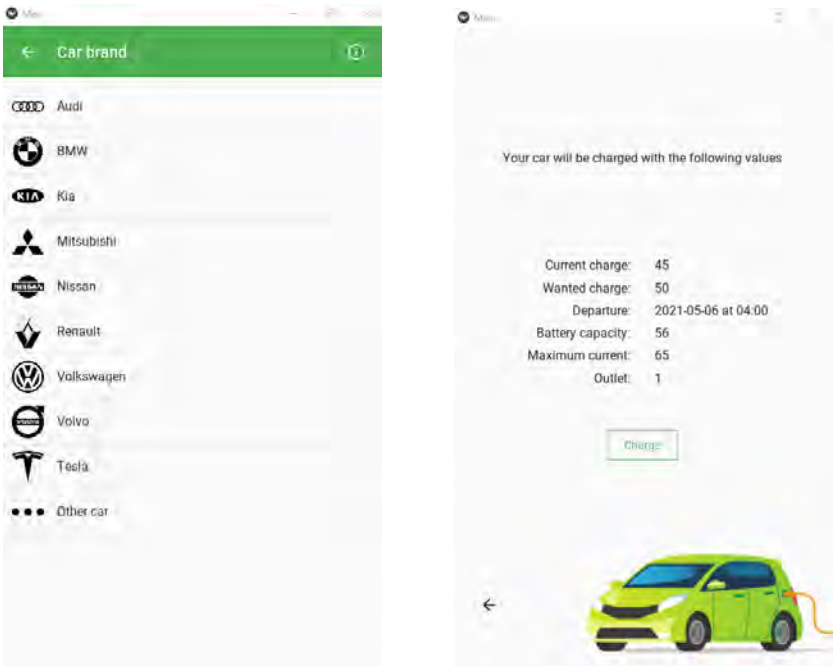
Med dessa buggar lösta kunde en datorsimulering ske på det sättet som beskrivs i 3.5.4. Under programmets körtid skrivs bilens förväntade procentuella laddning ut i konsolen samt den simulerade tiden. Därmed fastställdes det att backend:en fungerade som förväntat på simuleringsnivå. För att bevisa att backend:en fungerade på plats gjordes tester som beskrivs i 4.6.

4.5 Användargränssnitt

Projektets användargränssnitt består av 11 stycken skärmar, där bara en är synlig åt gången. En av skärmarna är villkorlig, och visas endast om användarens bil inte finns inlagd i systemet. På varje skärm finns ett antal komponenter i form av inmatningsfält, knappar eller etiketter. Ett exempel på två skärmar kan betraktas i Figur 4.7. Applikationens alla skärmar finns bifogade i bilaga .31.

De parametrar som användaren matar in i gränssnittet är; bilens nuvarande laddningsnivå, den önskade laddningsnivån vid avfärd, beräknad avfärdstid, bilens batterikapacitet och batteriets maximala tillåtna strömtillförsel. För att underlätta för potentiella användare som inte har specifikationerna på sin bils batterikapacitet och maximala strömtillförsel, så ges alternativet att välja sin bilmodell istället. Programmet tar ut motsvarande värden för bilen från en CSV-fil med de 28 vanligaste elbilarna i Sverige, som också kan uppdateras efter behov.

Ett allmänt lösenord används som identifiering på startsidan, detta för att förhindra att vem som helst ska kunna använda laddaren.



(a) Skärm med lista för bilmärken

(b) Skärm med summering av användarens inmatningar

Figur 4.7: Två exempel på skärmar i användargränssnittet

Användartestet som utfördes för användargränssnittet analyserades med hjälp av ”The thematic analysis method” där vissa mönster i resultaten noteras och delas in i så kallade teman. De svar av testanvändarna som pekade på samma användbarhetsproblem eller gav samma insikt delades in under samma tema, detta för att

koncentrera de kvalitativa resultaten till en grund för vidare utveckling. Följande teman kunde sammanställas av resultaten:

- Applikationen såg bra ut
  - Minimalistiskt
- Applikationen var lättanvänd och intuitiv
  - Positiv respons på den stegvisa användningen av inmatning
- För lite färgkontraster i vissa knappar
- Visst missnöje kring startsidan och förvirring kring vad applikationen innebär
- Vissa designfel där komponenter syns igenom tangentbord

I stora drag tyckte användarna att applikationen såg bra ut och var lätt att använda, mer specifika insikter var att applikationen var minimalistisk och den stegvisa inmatningen var lättanvänd. Vissa användbarhetsproblem uppmärksammades av testerna, som att vissa färgkontraster inte var tillräckliga, förvirring kring startsidan och designelement som syntes igenom andra. En full redovisning av användarnas svar finns i bilaga .1
Utifrån de resultat som samlats in gjordes några förbättringar på applikationen. På startsidan lades det till en kort text om vad programmet innebär och komponenter flyttades så att tangentbordet inte skulle täcka andra komponenter. De designfel som uppmärksammats åtgärdades också.

Utöver de problem som samlats in angående gränssnittets design med hjälp av testpersoner kunde följande anmärkas vid utvecklandet av användargränssnittet. Pekskärmens drivrutiner och dess samspel med Kivy fungerade undermåligt. Enheten behövde ytterligare konfiguration för att fungera i roterat läge, då pekkoordinaterna inte transponeras i samband med en mjukvarurotation. Även Kivys konfigureringsfil behövde anpassas för att känna igen pekskärmen som input. Trots dessa korrigeringar hade skärmen problem med att ens reagera på beröring, och ofta fel med en centimeter.

4.6 Tester

I projektet genomfördes totalt fem stycken tester för att kontrollera att produkten fungerade som den skulle samt för att möjliggöra felsökningar. Anledningen till att produkten testas ett flertal gånger var för att säkerställa dess funktionalitet för att kunna installeras på HSB Living Lab permanent.

4.6.1 Test 1 8/4-21

Under testet upptäcktes ett par buggar i programmet som togs bort. Dessa beskrivs under 4.4. Programmet reagerade på att bilen kopplades in i laddaren, men senare upptäcktes ett serverproblem som gjorde att förfrågningen om att starta laddningen inte kunde accepteras. Eftersom problemet var på serversidan avslutades testet.

Enligt API-dokumentationen skickas ett meddelande från CTEK:s server för att dubbelkolla att det är okej att starta respektive stänga av laddaren. Detta meddelande skickas till en server på den externa Raspberry Pi:n som förklaras i 3.5.3. Problemet var hur denna servern tog emot CTEK:s meddelande. Det var ett problem med hanteringen av JSON-objekt.

4.6.2 Test 2 12/4-21

Denna gången förväntades problemet på serversidan var löst. Vilket det borde varit. Problemet som upptäcktes denna gången var istället att servern aldrig tog emot CTEK:s meddelanden.
Utöver detta fel upptäcktes att laddaren får en Timeout om den inte är aktiv på 15 minuter. Detta ställde till problem då vi såg i optimeringsmodellen att vissa perioder skulle laddaren inte ladda bilen alls. Lösningen på problemet var att vid varje iteration som elbilen inte skulle laddas sattes en laddström på 0 A.

4.6.3 Test 3 20/4-21

Under detta testet förväntades det att CTEK skulle ha löst sitt problem med konfigurationen, dessvärre hade detta inte gjorts. Under detta testet förväntades det att serverproblemet skulle vara löst, dock visade sig det inte vara det. Efter kommunikation med CTEK fastställdes det att kommunikation med projektets server fungerade i det fallet meddelanden skickades manuellt. I det fallet meddelande skickades under laddning fungerade det stundvis men inte pålitligt.

4.6.4 Test 4 23/4-21

För att lösa serverproblemet togs beslutet att byta till en server som använts i ett tidigare test av detta projektet. Funktionaliteten i mjukvaran var detsamma dock var den programmerad i ett annat språk med andra bibliotek och den kördes på en annan sorts server. Genom att byta denna server fungerade nu programmet som det skulle och möjligheten till att ladda bilen mer effektivt hade nu skapats. Testet gjordes två gånger för att säkerställa programmets funktionalitet och det fungerade båda gångerna.

4.6.5 Test 5 28/4-21

Vid detta test kontrollerades att laddaren ”lyssnade” på de kommandon som skickades och inte endast laddade på maxkapacitet under hela tidsintervallet. Testet genomfördes med tre olika tillvägagångssätt, det första testet var att en tångamperemeter användes som kopplades kring en av ledarna i laddaren och ett testprogram kördes. I den första delen av detta test skickade testprogrammet att laddaren skulle skicka 9 ampere, därefter startades laddaren om för att genomföra samma test med skillnaden att laddaren skulle skicka 12 ampere, resultat hänvisas till bilaga .32 och .33.

Därefter gjordes ett till test med liknade tillvägagånssätt, ändringen som gjordes från det första testet var att testprogrammet skickade nu en signal som gjorde att laddaren skulle först ladda med 6 A i en minut, sen 12 A i en minut och därefter 9 A i en minut. När detta testet genomfördes upptäcktes även en startström på 14 A precis när laddaren startade. Resultat hänvisas till figur .34, .35, .36 och .37 i bilagor . Resultaten av mätningarna är även sammanställda i tabell 4.1.

Tabell 4.1: Strömmätning

Skickad ström (A)	Strömmätning (A)	Effektmätning (kW)
9	8.27	1.9
12	10.93	2.6
Test av olika strömstyrkor i en minut		
Startkommando	14.02	3.4
6	5.65	1.4
9	8.20	1.9
12	10.85	2.5

Efter detta gjordes även ett annat test för ytterligare kontroll av strömmen. Testet gick ut på att ladda en bil i 15 minuter med en ström på 6 A och sedan kontrollera den totala mängden kWh bilen hade laddat med under denna tiden. Nästa steg var att göra samma test men att istället skicka 12 A under lika lång tid, och det förväntade värdet var att i det andra fallet skulle värdet på den totala effekten vara ungefär dubbelt så stor. Detta test resulterade i följande, test ett gav 0.4 kWh och test två gav 0.7 kWh, vilket är ungefär det som förväntades.

4.6.5.1 Kontrollberäkningar

$$I = \frac{P}{\hat{U}} \tag{4.1}$$

$$\hat{U} = \frac{400}{\sqrt{3}} = 230.94V \tag{4.2}$$

Resultat från ekvationerna hittas i tabell 4.2.

Tabell 4.2: Beräknade strömmar jämfört med mätningar

Skickad Ström (A)	Strömmätning (A)	Beräknad ström (A)
Startkommando	14.02	14.72
6	5.65	5.63
9	8.20	8.26
12	10.85	10.83

5
Diskussion

5.1 Vad som användes

Den färdiga lösningen hade följande komponenter:

- Användargränssnitt, skrivet i Python med hjälp av Kivy och KivyMD
- Raspberry Pi Foundations 7” pekskärm
- 2x Raspberry Pi 4
- Optimeringsmodell, som tar hänsyn till spotpris, solkraftsprognos och belastningsprognos
- Omslutande chassi med kylning

5.2 Alternativ lösning

Den färdigställda prototypen, i form av en Raspberry Pi, med en pekskärm, och ett egendesignat chassi, visar att konceptet är genomförbart. Under arbetes gång hittades alternativa lösningar som hade resulterat i samma, och i vissa fall bättre funktionalitet. Vi kommer diskutera dessa lösningar i det kommande avsnittet.

5.2.1 Användargränssnitt och pekskärm

Användargränssnittet i sig fungerar väl, men att bygga stora delar av det med KivyMD var inte optimalt då ramverket fortfarande är i ett utvecklingsstadie. Till exempel är modulerna för datumväljare och tidsväljare bara tillgängliga i den senaste utvecklarversionen (KivyMD 0.104.2.dev0).

Den officiella pekskärmen från the Raspberry Pi Foundation fungerar bra för hemmaprojekt och för arbeten som ska visa koncept, men inte för en färdig produkt. De problem som förklarats tidigare, som problem med reaktion vid beröring och rotationsproblem gör att produkten inte uppfattas som professionell.

Som förbättring hade man kunnat bygga hela användargränssnittet som en webbapplikation, med ett ramverk som exempelvis ReactJS. Ett sådant ramverk hade förenklat uppskalningen av projektet, då alla enheter hade kommunicerat med en webbserver istället för att köra programmet lokalt. Till följd av detta hade funktions- och utseendeuppdateringar gått att sköta via webbservern, istället för att man antingen skickar ut tekniker till varje enhet, eller att man använder ett kommunikationsprotokoll som SSH för att styra uppdateringen av mjukvara.

En ytterligare fördel med att bygga en webbapplikation är att man inte är begränsad till en stationär skärm på plats, utan man kan nyttja de pekskärmar som folk bär med sig dagligen ändå, det vill säga mobiltelefoner. En stationär skärm löper större risk att vandaliseras, och måste även vara funktionell under Sveriges varierande väder, däribland skinande sol, men även regn, snö och kalla vintrar. Med detta i åtanke kan det rent ekonomiskt vara en bättre satsning att fokusera på en mobilanpassad webbapplikation, eller helt enkelt dedikerade Android och iOS-applikationer. Detta då stationära skärmar, samt de underhållstekniker som behövs, kommer att kosta mer än den server som behövs för en webb-baserad lösning. Den stationära skärmen skapar också en fråga om hygien, vilket kan väcka oro hos användarna, inte minst efter Covid-19 pandemin.

5.2.2 Chassi

Utöver det som nämns ovan om att använda en webb-applikation istället för en pekskärmslösning utgör chassit ett hinder. För att klara av miljön som laddstationerna är i krävs att chassit är vattentätt. Det innebär att chassit i stor utsträckning behöver vara slutet. Problemet blir då som nämnts i Metod att Raspberry Pi:en överhettar då den inte får tillgång till ny kall luft. Detta går att lösa antingen som beskrivits tidigare med en fläkt och ventilationshål men då offras vattentätheten. Ett annat alternativ är att använda kylflänsar som går genom chassit och ut utanför men detta skulle inte vara ekonomiskt hållbart.
Vid montering av chassit upptäcktes även att plasten som 3D-skrivaren använder sig av smälter ut mer än beräknat. Så inför en senare utskrift bör CAD-modellen uppdateras för att undvika att slipa chassit för att vara i lämplig storlek för skärmen.

5.3 Optimeringsmodell

De intentioner som projektet hade då elpris togs i beaktning var att det i regel följer samma trend som belastning på elnätet; är belastningen hög eller produktionen av el låg så stiger priset och vice versa med låg belastning. Ur marknadssynpunkt är detta också fördelaktigt då privatpersoner generellt sett engagerar sig mer i sin egna ekonomi än kring lasten på elnätet.
Optimeringsmodellen bygger i dagsläget på en blandning av prognoser och historiska medelvärden. Genom att införa en prognos även för pris/last på elnätet kan modellen förbättras ytterligare. De prisprognoser som hittades under projektets gång var tyvärr utom räckhåll med projektets begränsade resurser och kunde därför inte integreras.

Den största fördelen med modellen gentemot en vanlig laddstation är att den kan blicka 24 timmar framåt i tiden för att se när solpanelerna förväntas leverera tillräckligt hög effekt för att ladda. Detta innebär att belastningen på nätet minskar, men även att solpanelerna utnyttjas på ett effektivt sätt. Dessutom uppdateras modellen vid varje anrop från backend:en, vilket leder till att den alltid tittar 24 timmar

framåt och letar efter kraft från solpanelerna.

5.4 Prognoser

I projektet skapades både en lastprognos och en solkraftsprognos baserat på HSB Living Labs historiska värden. Att ta fram prognoser som dessa på egen hand är mödosamt, av lite olika anledningar. Solkraftsproduktion beror mycket på vädret, dels om det är soligt, men även temperatur påverkar verkningsgraden. För lastprognosen kan man heller inte helt korrelera datum och tid till förbrukad energi. Hur man lever i sitt egna hem är baserat på vanor, och även om det hade gått att förutsäga när en enskild individ förbrukar mycket energi blir det svårare när det är en större mängd, särskilt när folk flyttar ut och in i oregelbundna intervall. Istället för att då behöva ta alla parametrar i beaktning för att dimensionera och träna en modell, använde vi Rebase Energy för våra prognoser. Då Rebase är ett företag som har fokus just på AI-prognoser, och därmed har en underliggande plattform och modeller för att ta fram dem. Dessa modeller har då potential att förbättras med tiden, istället för att bli förlegade, som egendesignade modeller riskerar att bli.

5.5 Testresultat

I tabell 4.1 går det att se att det finns en korrelation mellan vad programmet ska skicka och vad laddaren faktiskt skickar för ström. Felmarginalen varierar mellan 5.83 % och 9.58 %, denna felmarginal kan vara beroende av många olika faktorer. En av förklaringarna är mätfel i instrument och program. När signalen skickas och tas emot av laddaren kan små fel komma med, som gör att strömmen inte blir samma. Dessutom finns det felmarginaler i mätaren som användes vilket också bidrar till felet.

Tabell 4.1 påvisar att ett mätfel har etablerat sig men den uträknade strömmen i dessa beräkningar har en mätfelskonstant som varierar mellan 6.17 % och 8.22 %. Skillnaden mellan de uträknade värdena i tidigare nämnda ekvationer och mätningarna som gjordes på plats skiljde sig inte mycket alls från varandra, variationen mellan dessa var 0.12 % och 0.18 % vilket tyder på att felet händer mellan programmet och laddaren.

Ett större problem med dessa mätningar är att det endast visades 2 värdesiffror på effektmätaren som var installerad i laddaren och denna visade två olika värden när 12 A skickades genom programmet, det styrker även argumentet att felet sker mellan programmet och laddaren. Felet här skiljer sig relativt mycket i fallet när mätaren visar 2.5 kW och 2.6 kW blir skillnaden mellan mätaren och uträkningarna 0.18 % i första fallet och 2.9 % i andra.

Ur det andra testet som gjordes förväntades det som tidigare nämnt att effekten skulle vara dubbelt så stor när strömmen var dubbelt så stor på grund av formel 4.1. Anledningen till att det inte gick att mäta exakt var att programmet som ska-

pats i ett tidigare projekt inte kollade mer än på en värdesiffra, det betyder att det finns en möjlighet att mätningarna var dubbelt så stora men det finns också risk för större fel. I och med att det finns lite resistans över alla ledare så är det också en faktor som behövs tas in i samtliga beräkningar.

5.6 Ett samhällsperspektiv

Figur 5.1: Personbilar i Sverige år 2020 uppdelat per drivmedel från trafikanalys. [20]

I dagsläget finns det 4 944 067 personbilar. [20] Av dessa är det endast elbilarna och laddhybriderna som kan laddas via elnätet. Det ger 178 024 bilar i Sverige som skulle kunna använda det vårt system i dagsläget. För att nå de minskade nivåer av utsläpp inom transportsektorn som landet har som mål till år 2030 krävs det en miljon laddbara bilar. [21] Detta skulle då leda till minskning av utsläpp på 70 procent. Ett hinder för att uppnå detta är infrastrukturen som tidigare nämnt i rapporten. Där kommer det här projektets mikrokontroller in som en teknisk lösning på problemet.

Vattenfall beskriver tio åtgärder som behövs ta för att uppnå målet av en miljon elbilar. Exempel på några av dessa åtgärder är miljözoner, både för personbilar men även lastbilar, ökat antal laddpunkter och incitament för företag att ställa om till elbilar. Mikrokontrollern skapar ett ytterligare incitament för aktörer på individnivå

samt kommersiell nivå att byta till en elbil. Både ur ett grönt perspektiv där man vill agera mer miljövänligt men även ur ett ekonomiskt perspektiv.

Som visas i figur 4.3 så minskas inte endast belastningen på elnätet utan även belastningen på plånboken. För ett företag med en bilflotta kan dessa summor växa och över en längre tid bli tillräckligt stora för att motivera ett företag till att använda mikrokontrollern. Företaget kan även ses som mer attraktivt. Både för en framtida kund men även för framtida arbetstagare. Att ett företag arbetar för att minska belastningen på elnätet är ännu en punkt att lägga till på listan med gröna initiativ ett företag arbetar med.

Detta gäller självklart även på en individnivå även om dessa summor eventuellt kan ses som försumbara för en individ kan exempelvis en bostadsrättsförening tillsammans införskaffa sig både laddpunkter samt mikrokontroller för att tillsammans minska sina utgifter och utsläpp.

Dock har mikrokontrollösningen sina nackdelar. Bland annat pris för produktion samt installation av dem. Där kan den alternativa lösning som beskrivs i 5.2 med en smarttelefon och en serverbaserad lösning fungera bättre. Där försvinner produktions- och installationskostnaden dock uppstår en ny serverkostnad. Då i princip varje person som påverkas av denna mikrokontroller äger en smarttelefon kan serverkostnaden vara värt det då materialet för mikrokontrollern ej behövs användas.

Oavsett vilken form lösningen tar så kommer den uppenbarligen inte vara tillräcklig för att själv lösa våra nuvarande klimatproblem. Däremot kan den ses som en knuff i rätt riktning och en uppmaning för att påskynda övergången till ett samhälle utan fossildrivna fordon.

5.7 Vid uppföljning av projekt

5.7.1 Användarkonton

Som nämnt i introduktionen implementerades inte användarkonton. Detta innebär att varje laddning blir anonym då den endast baseras på ett automatiskt genererat transaktions-id. Den enda säkerhetsåtgärden som existerar i programmet är ett lösenord som inte är kopplat till en viss användare utan det är kopplat till enheten som helhet. Därmed skulle vem som helst kunna använda mikrokontrollern. En uppföljning av projektet skulle därmed kunna implementera användarkonton. Dessa konton skulle kunna kopplas till en användares telefonnummer eller bilens registreringsnummer.

5.7.2 Integration av serverdel

I projektets gång användes en separat server, detta beskrivs i 3.5.3. Beslutet att separera servern togs tidigt för att undvika att behöva konfigurera om serverns IP-adress vid varje testtillfälle. Vid en enhetsdistrubition där mikrokontrollern blir installerad

på plats permanent är detta inget problem. Till följd av detta kan serverdelen integreras till att köra på mikrokontrollern. Detta minskar antalet enheter att felsöka samt att producera. Då kan man även utveckla produkten med det i åtanke och minska prestandaproblem som kan uppstå med att mikrokontrollern hanterar fler funktioner.

5.7.3 Livscykelanalys

För att få en bättre bild av produktens avtryck på miljön, samt slits vid verklig användning, bör en livscykelanalys genomföras. Med hjälp av denna data kan produkten vidareutvecklas för att få ett mindre miljöavtryck och längre hållbarhet.

6
Slutsats

Sammanfattningsvis så fungerar den färdiga produkten väl. Den löser uppgiften på det sätt som var avsett och minskar både belastning på elnätet, men även priset för privatpersoner som använder laddaren, vilket kan styrkas med de resultat som projektet tagit fram. På grund av den ökande marknaden för elektriska fordon behöver den maximala belastningen på elnätet minska vid specifika tider. Denna produkt har potential att bidra till att lösa det problemet.

Projektet går även att vidareutveckla till en universell produkt för andra märken på laddningsstationer för elbilar och också förenkla lösningen till en mobilapplikation eller liknande. En mobilapplikation beräknas lösa de prestandaproblem som har uppdagats under projektets gång. Lösningen med en mobilapplikation gör det även enkelt för användaren att snabbt skriva in sina parametrar, samt att ett användarkonto lätt kan kopplas till personen. Det finns dock fördelar med produkten om den installeras precis vid laddaren, eftersom det blir enkelt för användaren att komma ihåg att starta smartladdningen.

Kontroll över belastning på elnätet är något som kommer vara avgörande för att kunna integrera fler elfordon i det nuvarande elnätet. Denna typ av lösningar sparar också in kostnader för eventuella uppgraderingar på elnätet som kommer att behöva göras om fler elfordon ska användas.

Av testerna framgick att det finns ett mätfel som också diskuterats i tidigare kapitel. Under projektet var det ej relevant att arbeta med att minska detta fel då det inte var väsentligt för det slutgiltiga målet i detta skede. Slutsatsen som kan dras av testerna och resultaten från dem var att laddaren lyssnade på kommandona som skickades och sedan laddade bilen med en ström som är nära den som programmet bad om, det är det mest relevanta i och med att det är en av grundfunktionerna för optimeringsmodellen.

Modeller som denna kommer även vara en bidragande faktor till 100 % förnybar elproduktion, eftersom den använder sig av prognoser och kan vänta med till exempel elbilsladdning tills dess att produktionen från sol- och vindkraftverk är höga. Detta innebär förstås att prognosernas säkerhet kommer bli oerhört viktigt för en jämn belastning av elproduktionen.

Referenser

[1] A. A. och Daniel Kulin, "Elbilsläget 2018," 2018. URL: https://infogram.com/elbilslaget-2018-1h1749rjvkrq4zj?live.

[2] S. W. C. Östergren, "Elnätets beredskap inför en mycket storskalig utbyggnad av laddinfrastruktur i Sverige," 2019. URL: https://powercircle.org/wp-content/uploads/2020/05/2019-Exjobb-storskalig-utbyggnad-av-laddinfrastruktur.pdf.

[3] B. L. Annica Waleij Louise Simonsson, "Konsekvenser av energibortfall på samhällets funktionalitet och civilbefolkningens hälsa," 2019. URL: https://www.foi.se/rapportsammanfattning?reportNo=FOI-R--4755--SE.

[4] D. S. Anh Tuan Le Ali Fotouhi, "Innovative energy management system for smart buildings and grid interactions," 2019. URL: https://www.chalmers.se/en/projects/Pages/Innovative-energy-management-system-for-smart-buildings-andgrid.aspx.

[5] R. Larsson. URL: https://rl.se/vadret/gbgregn.php (hämtad 2021-05-06).

[6] D. Proctor, "Driving Change on the Grid - The Impact of EV Adoptiation," 2020. URL: https://www.powermag.com/driving-change-on-the-grid-the-impact-of-ev-adoption/.

[7] F. S. Fredrik Hedenus Martin Persson, *Hållbar utveckling - nyanser och tolkningar*. 2018, ISBN: 978-91-44-12187-1.

[8] M. Thorstensson, "Energiföretagen förklarar: Exportöverskott och ont om el – samtidigt," 2021. URL: https://www.energiforetagen.se/pressrum/pressmeddelanden/2021/energiforetagen-forklarar-exportoverskott-och-ont-om-el--samtidigt/.

[9] S. Energi, "Elåret verksamheten 2015," 2016. URL: https://www.energiforetagen.se/globalassets/energiforetagen/statistik/el/elaret/svenska-pdf/elaret2015_160429_web2.pdf.

[10] F. Martinsson, J. Gode, J. Arnell och J. Höglund, "Emissionsfaktor för nordisk elproduktionsmix," 2012. URL: https://www.ivl.se/download/18.343dc99d14e8bb0f58b7669/1445517637082/B2118.pdf.

[11] Energimarknadsbyrån, "Elpriser - prognos och utveckling," 2021. URL: https://www.energimarknadsbyran.se/el/dina-avtal-och-kostnader/elpriser-statistik/elpriser-prognos-och-utveckling/.

[12] Vattenfall, *Timpreiser på nordiska elbörsen*, 2021. URL: https://www.vattenfall.se/elavtal/elpriser/timpris-pa-elborsen/.

[13] D. Steen, 2019. (hämtad 2019-11-27).

[14] M. Nour, S. M. Said, A. Ali och C. Farkas, "Smart Charging of Electric Vehicles According to Electricity Price," i *2019 International Conference on Innovative Trends in Computer Engineering (ITCE)*, 2019, s. 432–437. DOI: 10.1109/ITCE.2019.8646425.

[15] The Raspberry Pi Foundation. URL: https://www.raspberrypi.org/products/raspberry-pi-4-model-b/.

[16] The Raspberry Pi Foundation. URL: https://www.raspberrypi.org/products/raspberry-pi-touch-display/.

[17] C. B. Jenifer Tidwell och A. Valencia, *Designing Interfaces : Patterns for Effective Interaction Design*. 2020, ISBN: 978-91-44-12187-1.

[18] J. Nielsen, "Why You Only Need to Test with 5 Users," 2000. URL: https://www.nngroup.com/articles/why-you-only-need-to-test-with-5-users/.

[19] Rebase Energy. URL: https://www.rebase.energy/.

[20] Trafikanalys. URL: https://www.trafa.se/ (hämtad 2021-05-07).

[21] Vattenfall, "Så når vi en miljon elbilar till 2030," 2018. URL: https://www.vattenfall.se/fokus/eldrivna-transporter/sa-nar-vi-en-miljon-elbilar-2030/ (hämtad 2021-05-07).

Appendices

Appendix I

	PERSON 1	PERSON 2	PERSON 3
OBSERVATION			
TASK 1.			
TASK 2.			
TASK 3.			
TASK 4.			
INTERVIEW			
QUESTION 1.	Snygg design	Gillade designen, enkelt	Enkelt att använda, nice
QUESTION 2.	Lätt att använda	Gillade lösenord och lättanvänd	-
QUESTION 3.	Nej	Mer färg, kanske gult?	-
QUESTION 4.	Kanske om ens bil inte fanns med som alternativ	Nej enkel att använda	Nej
QUESTION 5.	Nej	Nej	Nej
QUESTION 6.	Jättebra och smidigt!	Imponerad	Snyggt

Figur .1: Svar från användartest

	PERSON 4	PERSON 5	PERSON 6
OBSERVATION			
TASK 1.			
TASK 2.			
TASK 3.			
TASK 4.			
INTERVIEW			
QUESTION 1.	Intuitiv, minimalistisk (bra)	"vad ska jag göra nu?" oklart i början. Minimalistisk, men otydlig	Den va clean
QUESTION 2.	Stegvis inmatning	Straight forward, inte mycket att göra fel på	Lätt att använda, användarvänlig
QUESTION 3.	Bättre färgkontraster, ex knapparna	Småfel i designen (knappar syns igenom tangentbordet på första	Om man ej kan sina batterispecs
QUESTION 4.	Nej	Ladda med optimeringsmodell, vad betyder det?	Nej
QUESTION 5.	Användarsystem, där appen har koll på ens bilar	Tidigarenämnda småsaker, varna innan sista sidan att info saknas	-
QUESTION 6.	Det va fint, jobbigt att mata in lösenordet varje gång	Straight forward, förklara opt-modell, välja vad man optimerar för (sol, belastning, pris)	Roligare förstaskärm

Figur .2: Svar från användartest

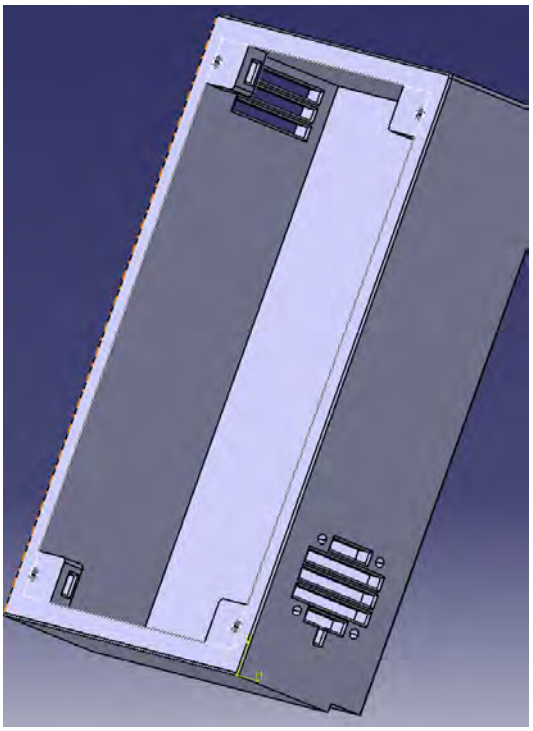
	PERSON 7	PERSON 8
OBSERVATION		
TASK 1.		
TASK 2.		
TASK 3.		
TASK 4.		
INTERVIEW		
QUESTION 1.	Fin, nice	Mycket vitt
QUESTION 2.	Lätt att använda	En fråga i taget
QUESTION 3.	Nej	Kivymd fungera ej på pycharm annars bra
QUESTION 4.	Other istället för other car (Otroligt minor inget att bry sig om)	Nej
QUESTION 5.	-	Vad sparar jag? (Förstår att det är svårt att implementera)
QUESTION 6.	Fett snyggt	Nej

Figur .3: Svar från användartest

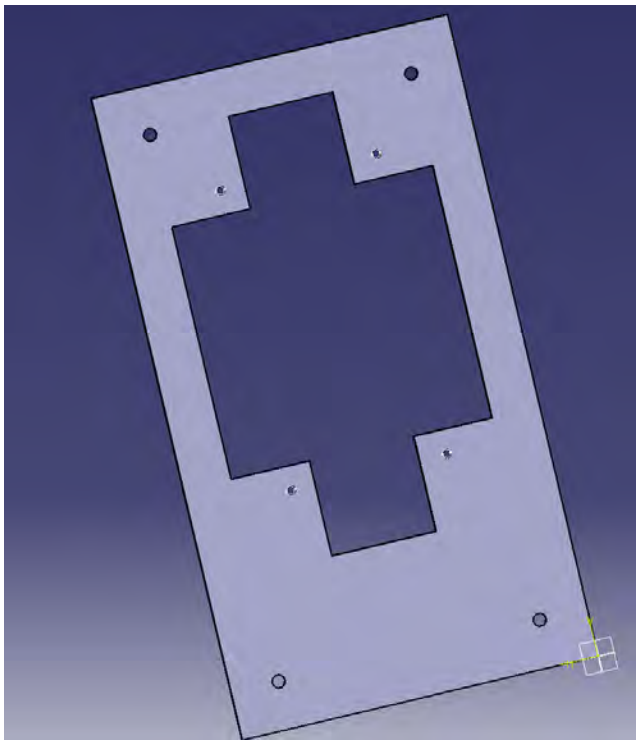
```
1. Welcome user
2. Reassure/sign written compliance form
3. Explain a little about the project
    - The object is to test our user interface for our smart charging station.
    - The charging station is running an algorithm that computes when to charge to lower the cost and minimise load on the electric network at peak load times.
    - The values you put in will be put into our algorithm.
    - We will observe when you complete a few tasks and afterwards we will have a small interview.
4. The password is: 80085

OBSERVATION------------------------

    Task 1.
        You want to charge your car (Tesla Model S) with the charging station. Use the touch interface to start charging the car.
            Parameters to put in:
                -Current charge: 88%
                -Wanted charge: 100%
                -Departure Date & Time: Today at 20.00
                -Car: Tesla Model S
                -Outlet: 1

-- reset

    Task 2.
        You want to charge your car, but your car model is not in the list of cars. Use the touch interface to start charging the car.
            Parameters to put in:
                -Current charge: 88%
                -Wanted charge: 100%
                -Departure Date & Time: Today at 20.00
                -Battery capacity: 100
                -Maximum current: 16
                -Outlet: 1

-- reset

    Task 3.
        You want to charge your car and no paramets are given. Use the touch interface and external tools to start charging the car. You can choose how much charge percentage you want at departure and time of the departure yourself.

-- reset

    Task 4. You want to charge your car, but you don't want to use the optimisation algorithm this time. Use the touch interface to start charging your car.

INTERVIEW--------------------------

    1. What were your first impression of the application?
    2. Can you name something you liked about the application?
    3. Can you name something you disgiliked about the application?
    4. Were anything confusing in the application? If yes, what?
    5. Do you have any suggestions for improvement in the development of the application?
    6. Comments?
```

Figur .4: Manus användartest för användargränssnittet

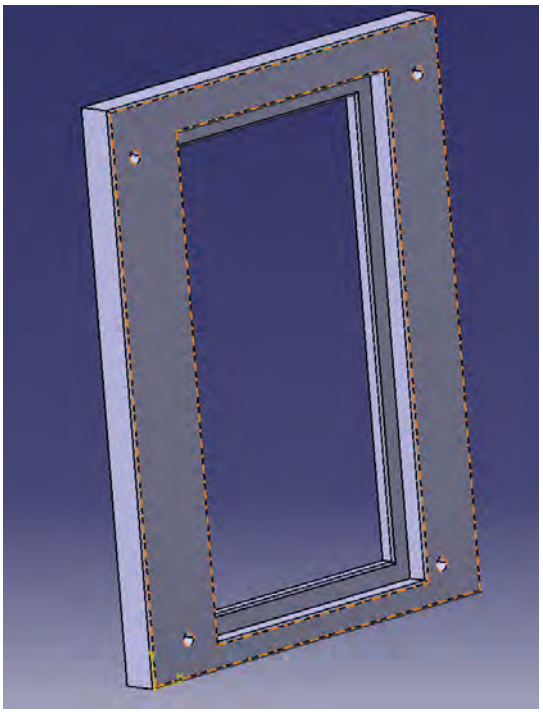
Appendix II



Figur .5: Huvuddel av chassit

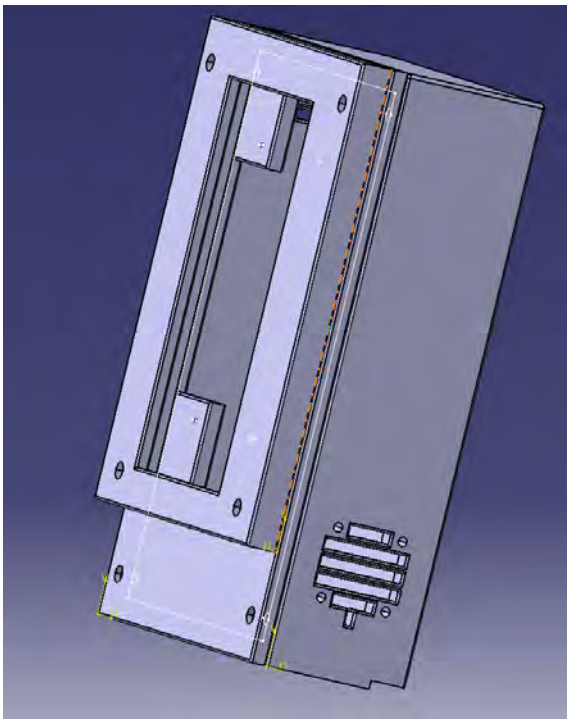


Figur .6: Mellanlägg till chassit



Figur .7: Lock till chassit

f



Figur .8: Hela chassit

Figur .9: Den färdiga produkten

h

Figur .10: Produkten sedd från höger, med luftintag synligt

Figur .11: Produkten sedd från vänster, med utblås synligt

j

Appendix III

Figur .12: Inmatning för lösenkod

Figur .13: Fel lösenord

Figur .14: Inmatning för nuvarnade batteriprocent

l

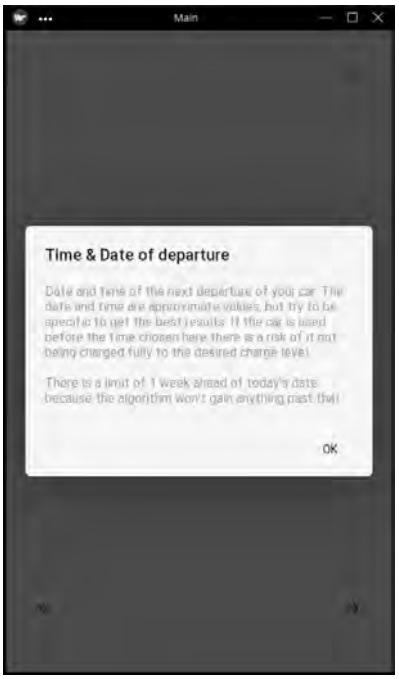
Figur .15: Informationsruta för nuvarande batteriprocent

Figur .16: Inmatning för önskad batteriprocent

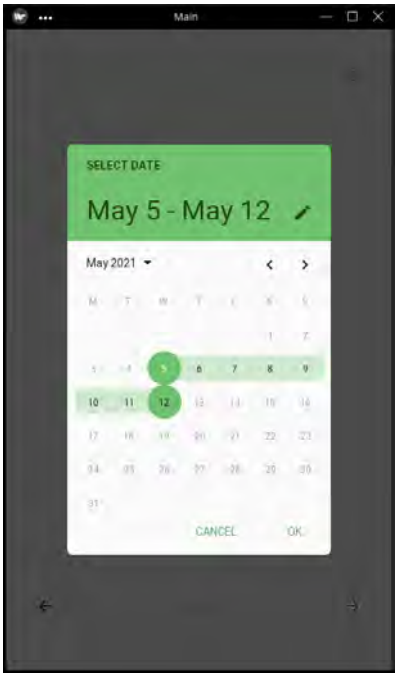
Figur .17: Informationsruta för önskad batteriprocent

Figur .18: Välj datum och tid

n



Figur .19: Informationsruta om hur datum och tid väljs, samt tidsbegränssningar



Figur .20: Datumväljarverktyget

Figur .21: Tidsväljarverktyget

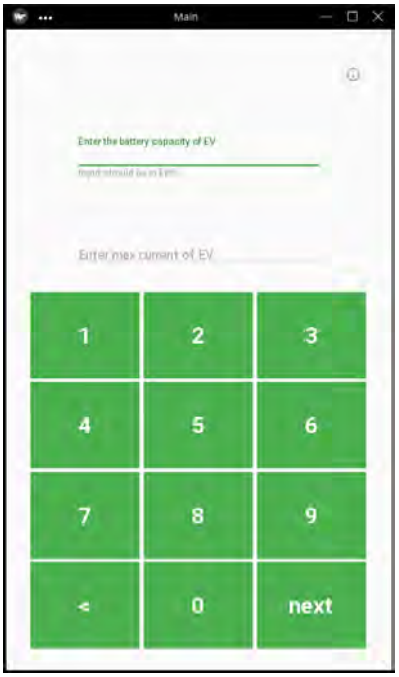
Figur .22: Hur det ser ut när datum och tid är vald

p

Figur .23: Bilmärkesmeny

Figur .24: Modellmenyn efter att bilmärke har valts

Figur .25: Informationsruta om bilmärkesmenyn



Figur .26: ”Other”-menyn, där självinmatning av batterikapacitet och maximal strömstyrka sker

r

Figur .27: Informationsruta om batterikapacitet och maximal ström

Figur .28: Välj laddaruttag

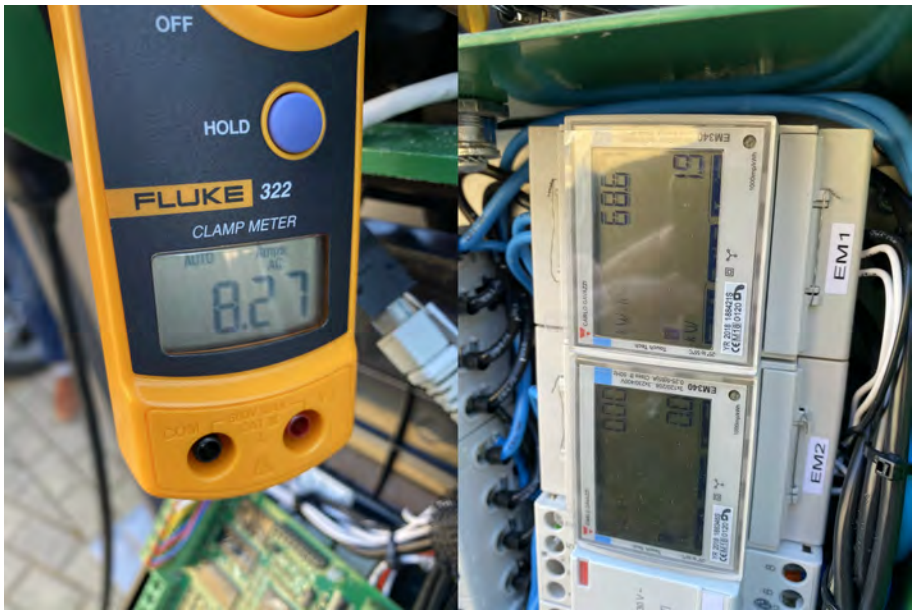
Figur .29: Informationsruta om laddaruttag

Figur .30: Sammanfattningsruta

t

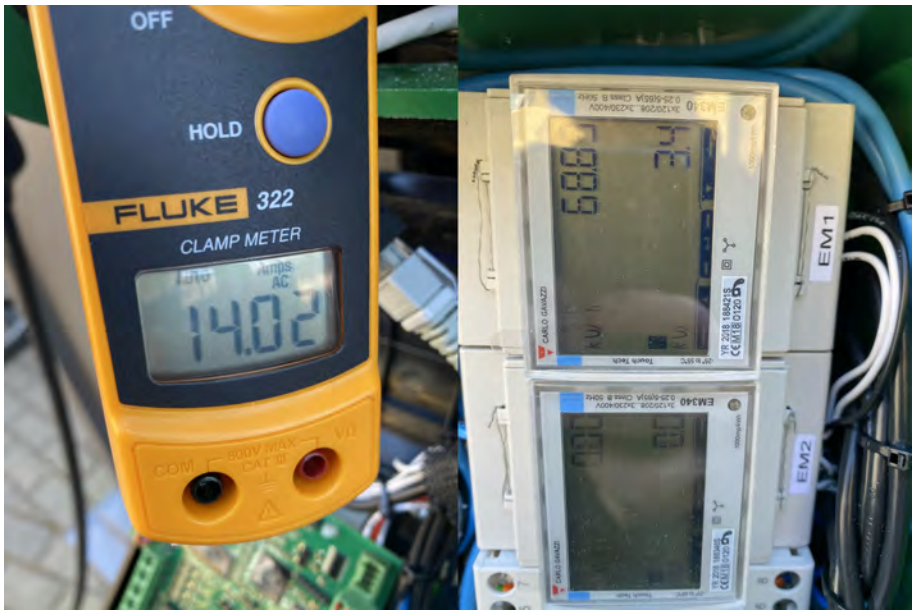
Figur .31: Programmet förbereds för att ta emot nästa användare

Appendix IV

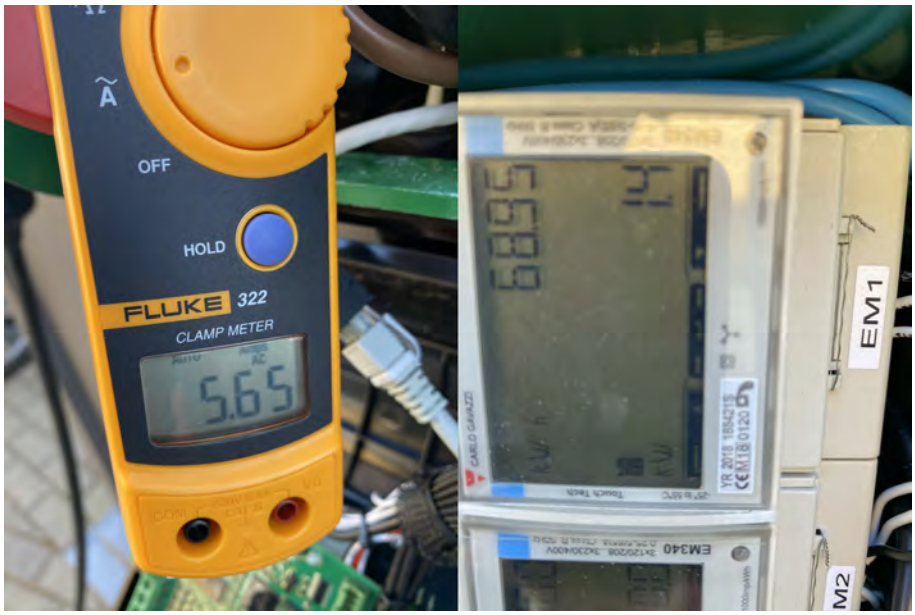


Figur .32: Första test av 9 A

Figur .33: Första test av 12 A



Figur .34: Ström vid uppstart av laddaren



Figur .35: Test med 6 A strömstyrka i en minut

Figur .36: Test med 12 A strömstyrka i en minut

Figur .37: Test med 9 A strömstyrka i en minut

INSTITUTIONEN FÖR ELEKTROTEKNIK
CHALMERS TEKNISKA HÖGSKOLA
Göteborg, Sverige
www.chalmers.se

CHALMERS