SepiSolar™
POWER BY DESIGN

High-level Software Design

Prepared by: Dave Jarvis
Prepared for: Josh Weiner & Pragash Kothandaraman
April 29, 2019

Contents

1 Overview

This document describes a high-level software design resulting from an open discussion with Benjamin Gottheil and John Henley.

1.1 Purpose

This document defines high-level software requirements for an energy storage modeling system. The system is intended to simplify the workflow that solar contractors use to determine whether a potential customer offers a viable business opportunity for an energy storage solution. Key determination factors include:

- What size of energy storage is required?
- What is the solution's return on investment (ROI)?

1.2 Audience

Readers should be familiar with software development process, including database design, component diagrams, and use cases.

1.3 Stakeholders

The stakeholders include:

- Josh Weiner
- Pragash Kothandaraman
- John Henley
- Benjamin Gottheil

1.4 Motivation

Currently a spreadsheet is used to brute force optimal system specifications to maximize the ROI for energy storage solutions. This process is time-consuming and requires in-house expertise. Migrating the spreadsheet to a web application aims to significantly improve performance while providing solar contractors with the ability to generate their own reports for decision-making.

1.5 Scope

The following table lists items inside and outside of scope for the first release:

Description	In Scope?
Mobile app	N
Integrated payment system	N
Internationalization	N
Interactive graphs	N
Account administration and registration	N
Group administration	N
Systems monitoring and failure notification	N
Third-party application failure notifications	Y
Users and groups	Y
Authentication and authorization	Y
Energy usage graphs	Y
Intelligent electricity allocation	Y
Tariff tracking	Y

Table 1.1 Application scope

2 Definitions

The following terms, acronyms, and abbreviations are used within the document:

API	Application Programming Interface
Breadcrumbs	User interface widget used for navigation
CSS	Cascading style sheets
CSV	Comma-separated values
ESS	Energy storage system
ERD	Entity-relationship diagram
HTML	Hypertext Markup Language
HTTPS	Secure hypertext transport protocol
JVM	Java Virtual Machine
kB	Kilobyte (1000 bytes)
KKT	Kuhn-Karush-Tucker (check conditions for a function's minimum)
kWh	Kilowatt hour
PII	Personally identifying information
Purchaser	A solar contractor's customer
PV	Photovoltaic
Region	State or Province
ROI	Return on investment
Solar Contractor	A business involved with installing energy storage and solar systems
SQL	Structured Query Language
UAT	User acceptance testing
UI	User interface
Utility	A business that sells electricity to property owners

3 Architecture

This section provides a high-level overview of how the system responsibilities are partitioned into components.

3.1 Component Diagram

The architecture diagram depicts how the system is decomposed and how major components relate. Major architectural components are shown in the following diagram:

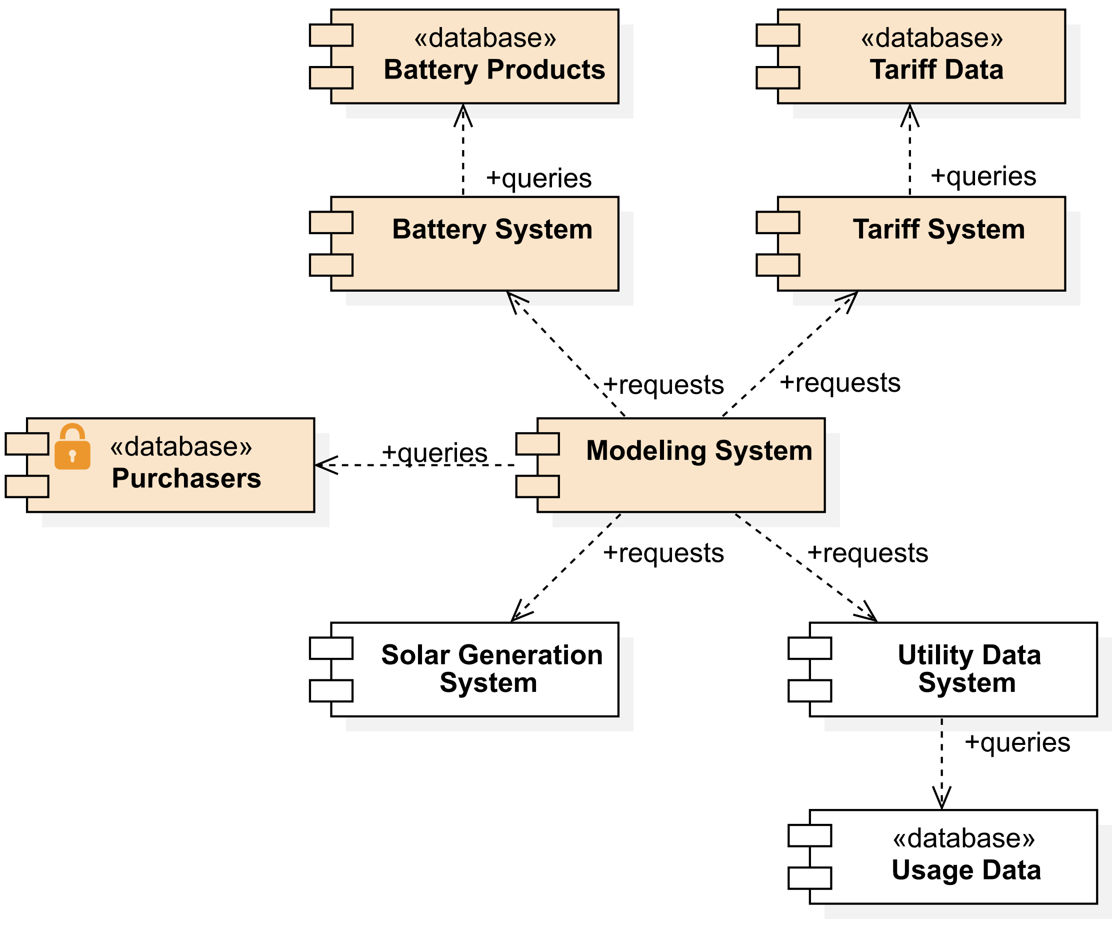


Figure 3.1 Architectural Components

The highlighted components are developed internally; the **Modeling System** is a net new application to replace an existing spreadsheet solution.

3.2 Internal Interfaces

The **Modeling System** component is the main application. It is responsible for acquiring user input, requesting data from external systems, storing personally identifying information (PII), and generating a comparative analysis report. Further, a subsystem will determine the optimal time to allocate incoming electricity for charging batteries based on tariff data, thereby minimizing purchasers' costs.

The **Purchasers** component is a new relational database that stores a person's contact information, address, electricity data usage, and energy storage system (ESS) configuration. The usage data for a purchaser is populated by and queried by the **Modeling System**. No other components may query the database.

The **Battery System** component provides a simple application programming interface (API) to query batteries and their properties. Broadly, the system can receive (1) queries to find batteries matching a set of criteria; and (2) queries for a specific battery's properties. This component exists.

The **Tariff System** component provides an API to query the daily tariffs applied by a utility company when charging their customers for electricity usage. Broadly, the system can receive queries to find a list of available utility companies, the tariffs for a specific utility company, and any qualifiers that apply to a particular tariff. Additionally, the system can return a list of electricity costs for a given date, in 15-minute intervals. This component exists.

This list is not exhaustive; additional components could include: graph generation, report generation, and a photovoltaic system.

3.3 External Interfaces

The **Solar Generation System** component is a third-party application that can generate the estimated energy output for a photovoltaic (PV) system given specific parameters. The parameters include location information (i.e., zip code), tilt angle, mounting type, and array size in kilowatt hours (kWh). This component exists.

The **Utility Data System** component is a third-party application that can relay electricity usage data for a given building, sourced from the appropriate utility company. The **Modeling System** queries this component by providing (1) the purchaser's address; and (2) the contact information of the recipient who will receive the purchaser's electricity usage data. This component exists.

4 Use Cases

This section describes how users may interact with the system to obtain the information they have requested.

4.1 Actors

The system has the following actors:

- **Battery System** – An internal system that can store, retrieve, and search for battery properties.
- **Recipient** – Solar contractor who receives electricity usage data requests.
- **Solar Generation System** – A third-party system that can generate expected kWh for a PV system based on various parameters.
- **Tariff System** – An internal system that can store, retrieve, and search for utility company tariffs.
- **User** – Salesperson who wants to determine whether a purchaser is a viable business lead for a solar contractor.
- **Utility Data System** – A third-party system that can request electricity usage data for a purchaser.

4.2 UC001 – Generate Report for New Lead

A **User** generates a report for a solar contractor on behalf of a prospective purchaser.

4.2.1 Actors

The following actors are involved in this use case:

- **Battery System**
- **Recipient**
- **Solar Generation System**
- **Tariff System**
- **User**
- **Utility Data System**

4.2.2 Diagram

The following diagram provides a high-level overview of this use case. In the diagram, it is apparent that obtaining electricity usage data is outside the purview of the modeling system.

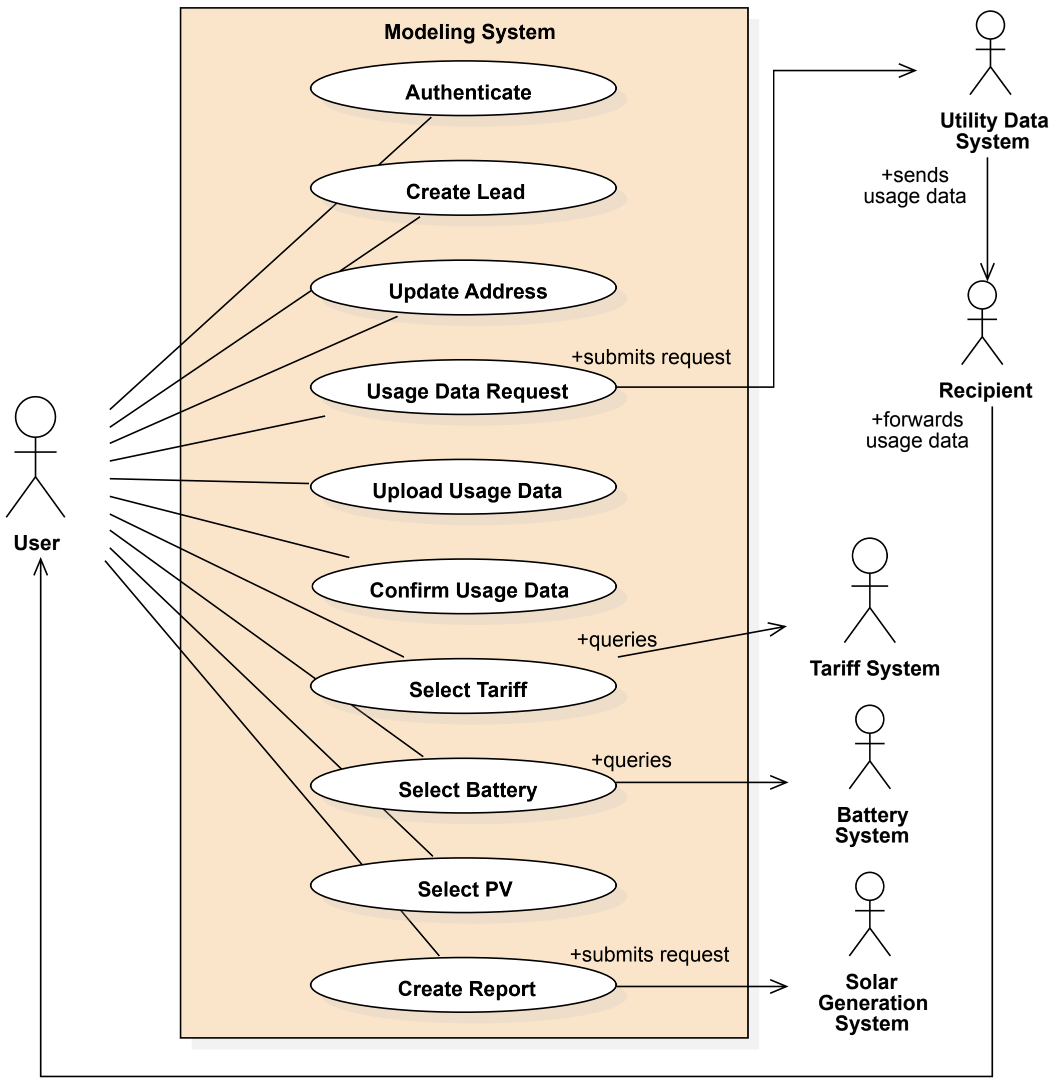


Figure 4.1 Use Case 001

4.2.3 Preconditions

The following preconditions are required to complete this use case:

- All databases are online.
- All third-party systems are online.
- The web application server has sufficient memory to complete the requests.
- **User** has an account in the energy savings performance modeling system.
- **User** has a copy of purchaser's electricity usage data.
- The electricity usage data is complete (i.e., has no gaps).

4.2.4 Postconditions

The following postconditions are met upon successful completion of this use case:

- Purchaser's PII is stored, securely.
- Purchaser's electricity usage data is stored.
- The **User** receives an energy savings report.

4.2.5 Main Flow

The main flow of this use case follows:

1. **User** browses to energy savings performance modeling system.
2. **User** authenticates using email and password for existing account.
3. **User** opts to create a new lead.
4. **User** fills out form for contact and address information.
5. **User** selects recipient to receive electricity usage data (not shown).
6. **System** stores contact and address information.
7. **System** presents confirmation page to **User**.
8. **User** confirms the information is correct.
9. **System** submits request for electricity usage data to **Utility Data System**.
10. **Recipient** receives electricity usage data.
11. **Recipient** sends electricity usage data to **User**.
12. **User** uploads usage data.
13. **User** reviews usage data.
14. **User** selects the utility company and corresponding tariff.
15. **User** selects battery configuration.
16. **User** selects photovoltaic parameters (not shown).
17. **User** submits request to generate report.
18. **System** submits photovoltaic parameters to **Solar Generation System**.
19. **System** receives daily kWh energy from **Solar Generation System**.

20. **System** calculates optimal energy storage usages based on tariff rates.
21. **System** generates an energy savings report.
22. **User** receives the energy savings report.

4.2.6 Alternative Flows

To be discussed.

4.2.7 Exceptions

In steps 9 or 18, if the third-party system goes offline:

1. The **User** receives an error message indicating that report creation is unavailable.
2. The **System** notifies the application administrator that report creation is offline.
3. The **System** enters "offline mode" until an administrator resolves the issue.

5 User Interface

This section defines the general application flow using wireframes. The web application will use responsive CSS, but be developed for resolutions of 1024x768.

5.1 Login

The login page authenticates users and is the first page shown by the application.

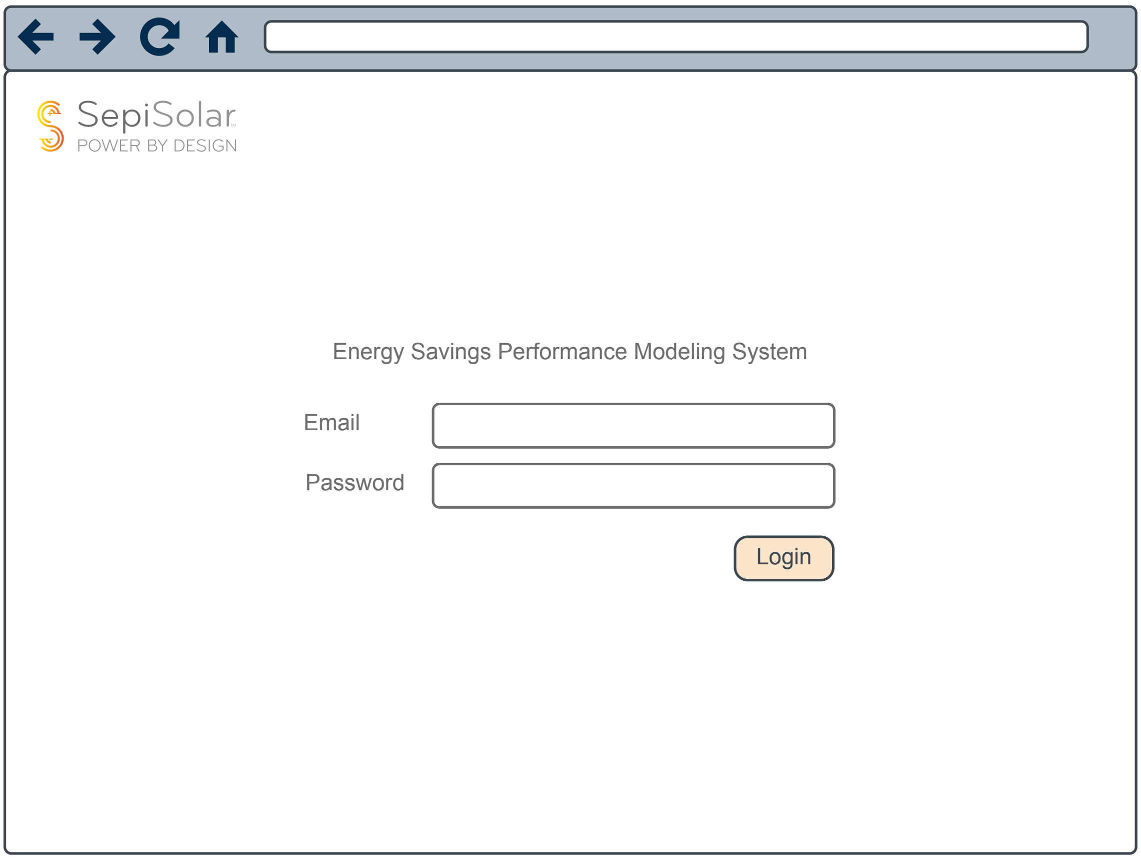
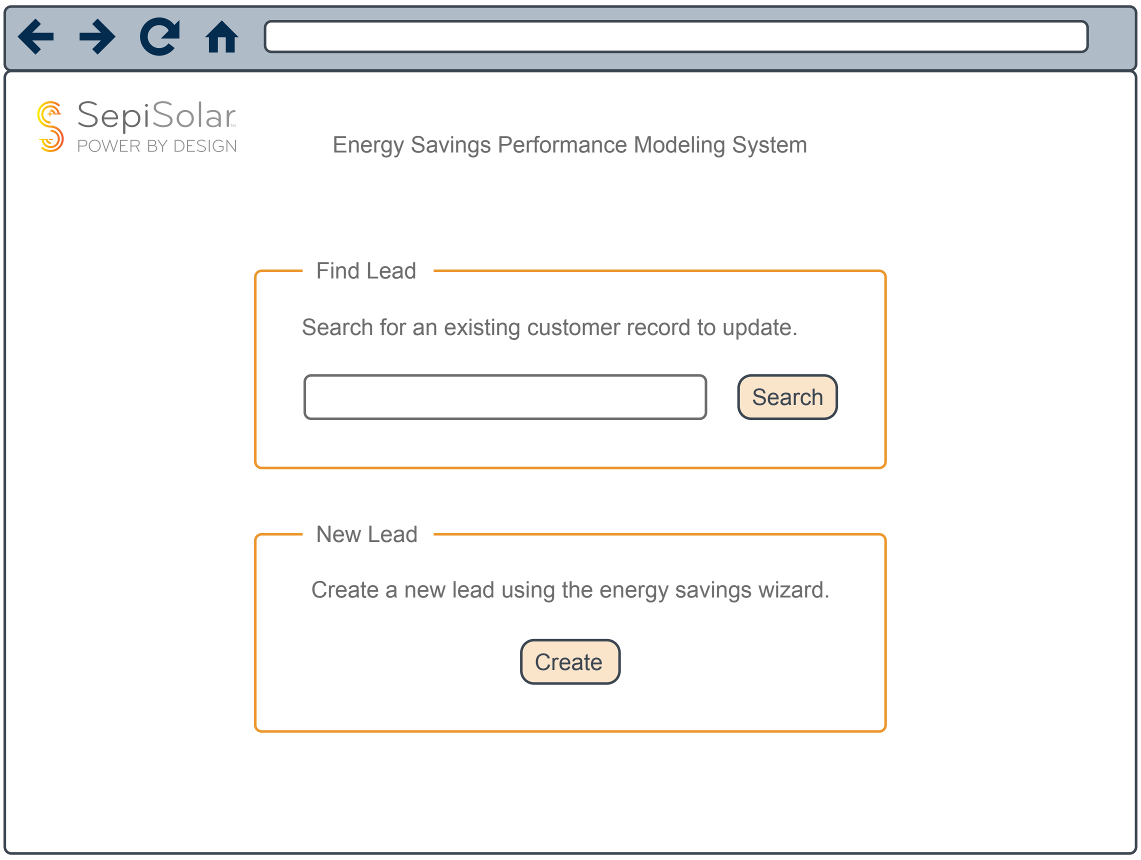


Figure 5.1 Login Page

5.2 Home

The home page for a salesperson is displayed when authentication is successful. When invalid credentials are supplied an error message is displayed. When an excessive number of attempts are made the user is locked out of their account.


SepiSolar
POWER BY DESIGN

Energy Savings Performance Modeling System

Find Lead

Search for an existing customer record to update.

Search

New Lead

Create a new lead using the energy savings wizard.

Create

Figure 5.2 Home Page

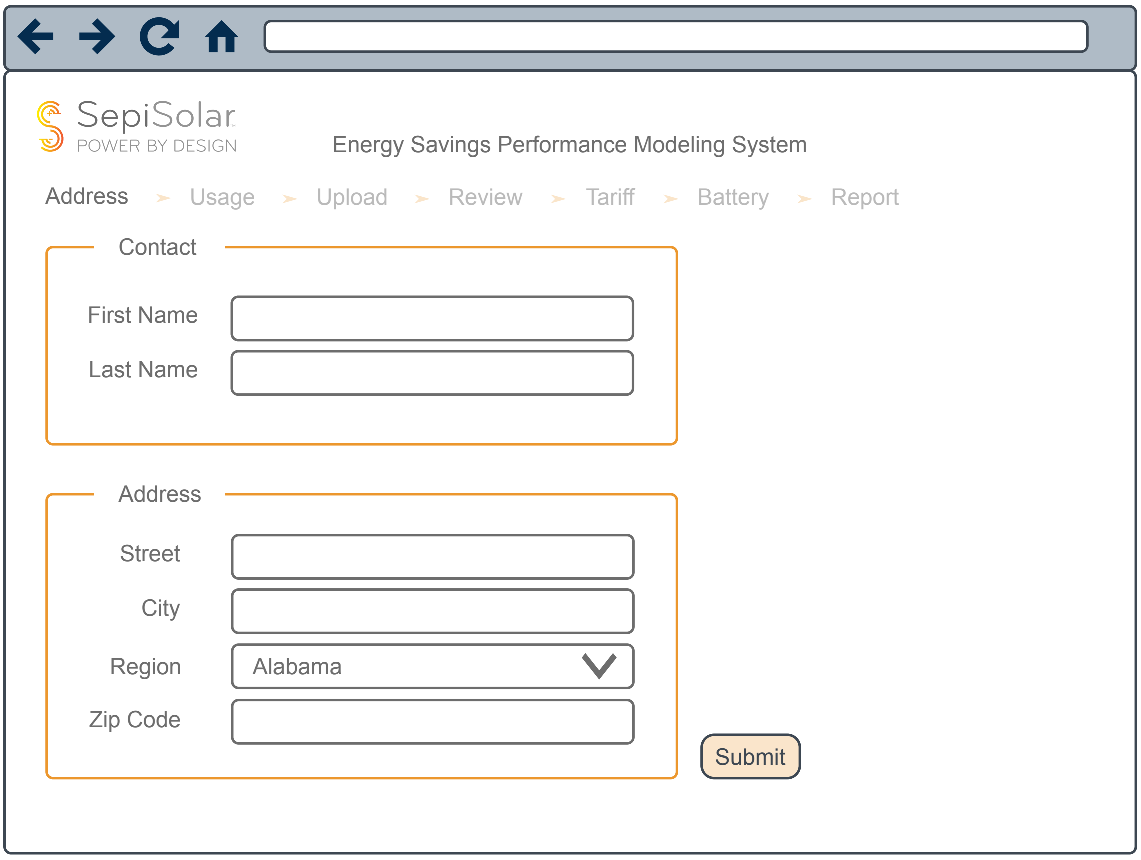
The home page for an administrator would include additional options for maintaining users and groups.

5.3 Wizard

When users opt to create a new lead (i.e., a purchaser), a wizard guides them through the process. Users can return to any previous page via the breadcrumbs.

5.3.1 Address

The first page prompts users to provide contact information for a new lead. More fields may be required than those shown, depending on the detailed requirements.



Contact

First Name	
Last Name	

Address

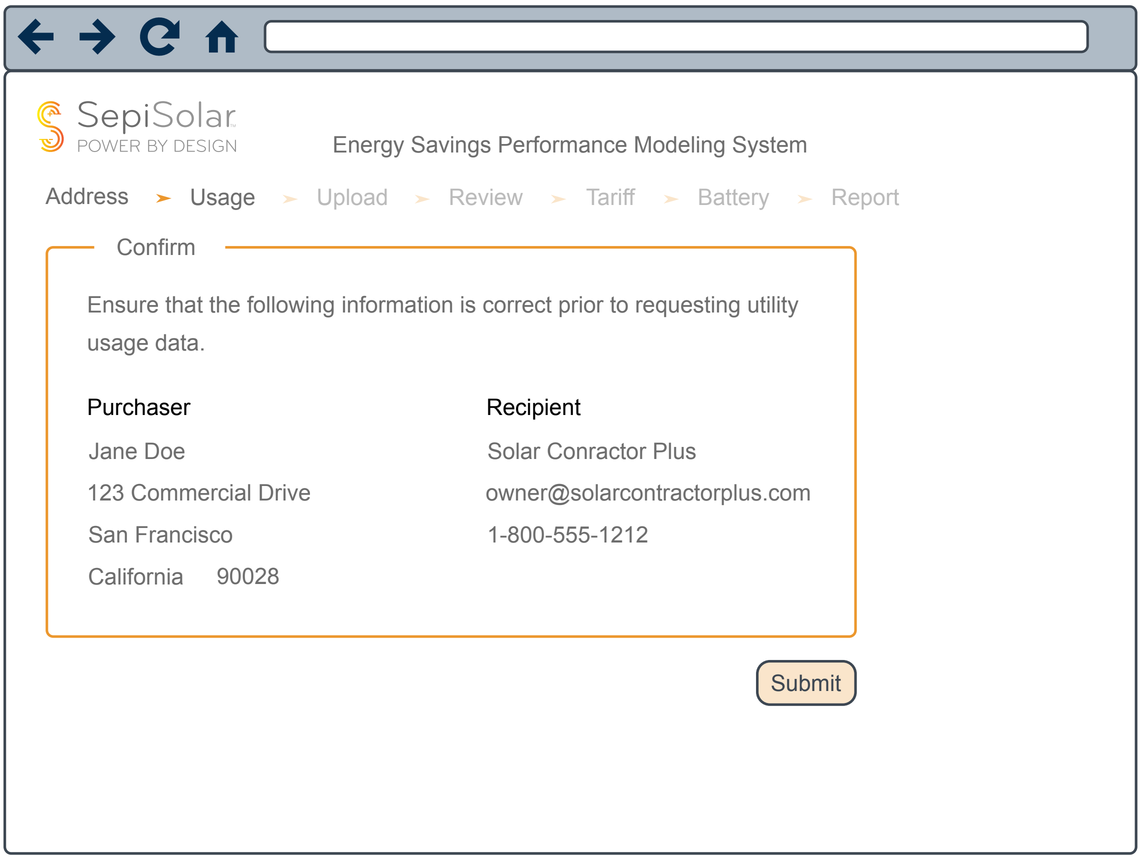
Street	
City	
Region	Alabama
Zip Code	

Figure 5.3 Wizard Page – Address

The contact information is necessary to obtain electrical usage data for the given address from the relevant utility company. The utility company name may also be required.

5.3.2 Usage

The second page prompts users to issue electricity usage data requests. The recipient information indicates who will receive the information. The form for entering the recipient information is not shown, but would be similar to the address page.



SepiSolar POWER BY DESIGN

Energy Savings Performance Modeling System

Address ➢ Usage ➢ Upload ➢ Review ➢ Tariff ➢ Battery ➢ Report

Confirm

Ensure that the following information is correct prior to requesting utility usage data.

Purchaser	Recipient
Jane Doe	Solar Conractor Plus
123 Commercial Drive	owner@solarcontractorplus.com
San Francisco	1-800-555-1212
California 90028	

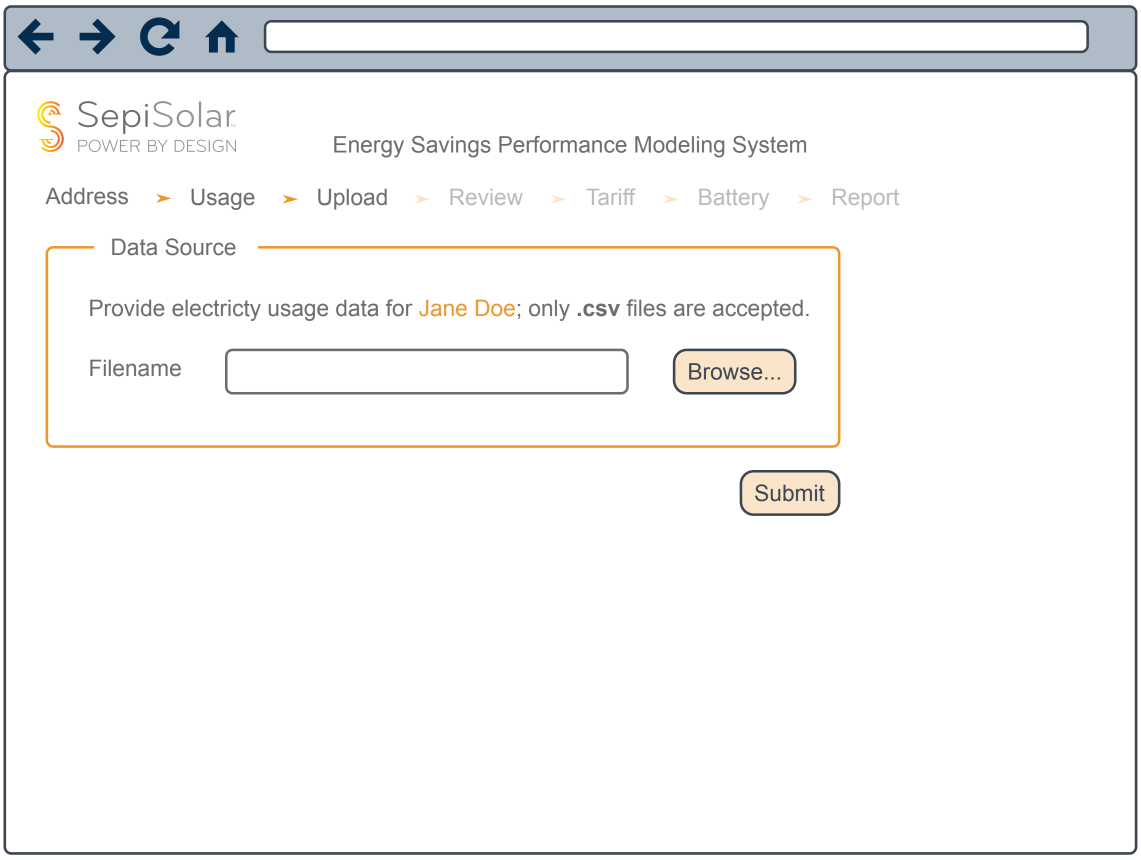
Figure 5.4 Wizard Page – Usage

5.3.3 Upload

Electricity usage requests take time to process by utility companies. When the electricity usage data is received, users:

1. Return to the application.
2. Authenticate, if required (depending on session expiry).
3. Find and select the installation location (i.e., contact information).
4. Select the installation location from a list.

The system then returns to the wizard and prompts users to select and upload the appropriate data file.



SepiSolar POWER BY DESIGN

Energy Savings Performance Modeling System

Address ➢ Usage ➢ Upload ➢ Review ➢ Tariff ➢ Battery ➢ Report

Data Source

Provide electricty usage data for Jane Doe; only **.csv** files are accepted.

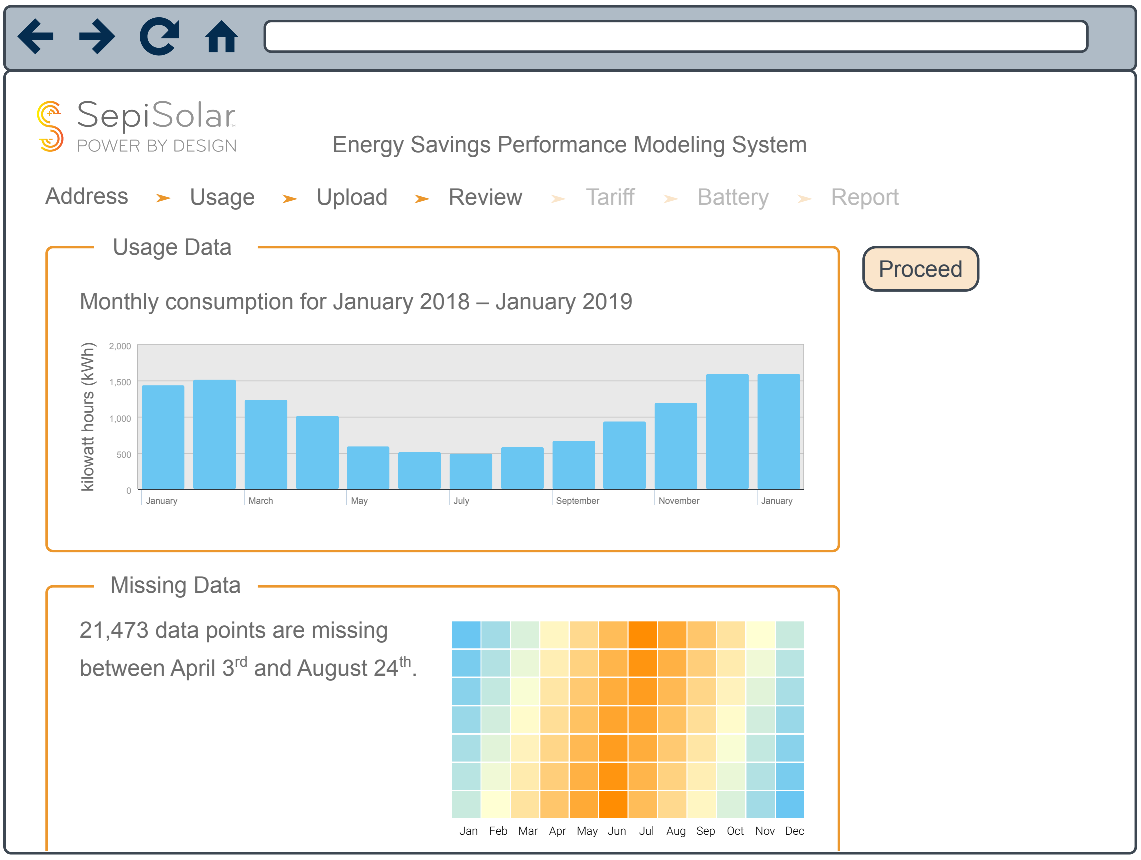
Filename

Figure 5.5 Wizard Page – Upload

In future versions, it may be possible to automate integration of the data request with the system itself, thereby skipping this step.

5.3.4 Review

The uploaded usage data may be insufficient to perform a comparative analysis for the ROI. Users are prompted to review the usage data to check for anomalies. Large numbers of data gaps can be represented as a heat map to reveal patterns. Small data gaps can be listed in tabular format (not shown).



Energy Savings Performance Modeling System

Monthly consumption for January 2018 – January 2019

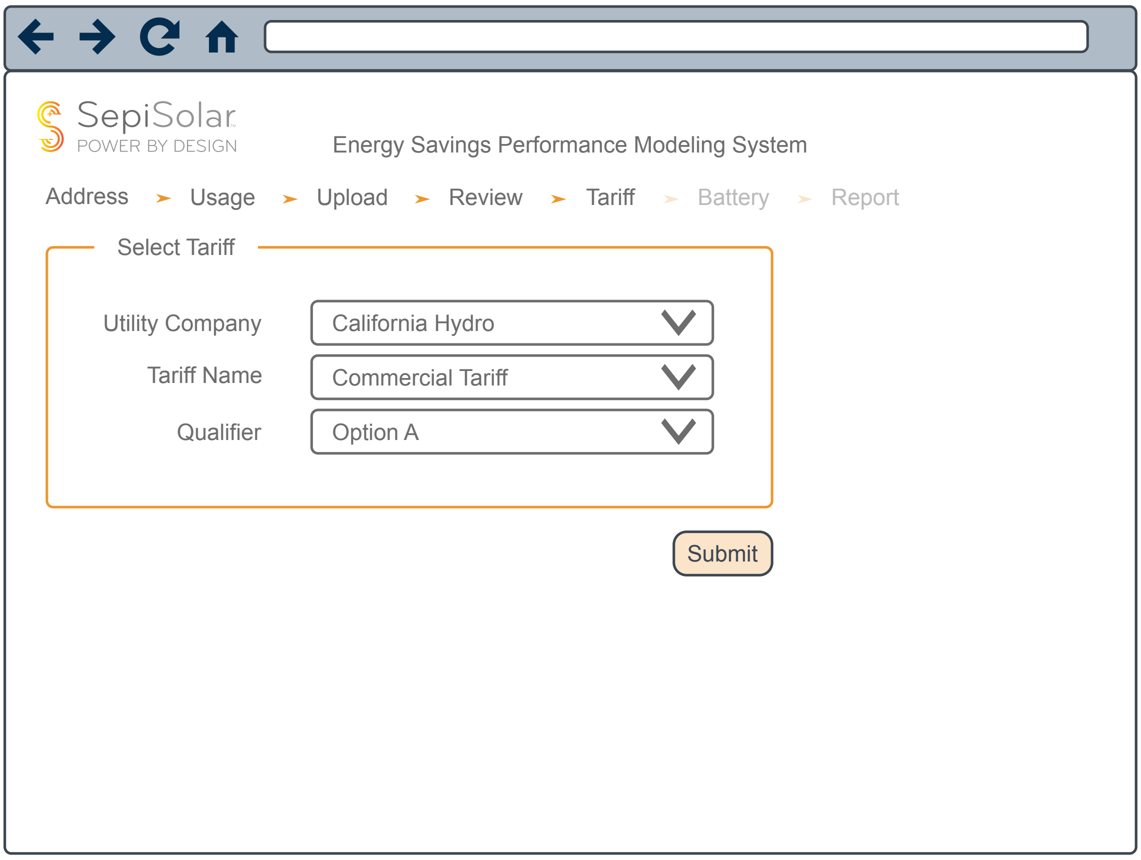
21,473 data points are missing between April 3rd and August 24th.

Figure 5.6 Wizard Page – Review

If users are satisfied with the amount of missing data, they may continue. Otherwise, they may return to any previous page using the breadcrumbs. This affords users the possibility of re-submitting a usage data request or uploading a new data file.

5.3.5 Tariff

Different utility companies charge different rates for electrical usage at different times of the day. This page allows users to select the appropriate tariff that will lead to an equitable comparison between electricity from the grid and electricity from an energy storage solution.


SepiSolar POWER BY DESIGN

Energy Savings Performance Modeling System

Address ➤ Usage ➤ Upload ➤ Review ➤ Tariff ➤ Battery ➤ Report

Select Tariff

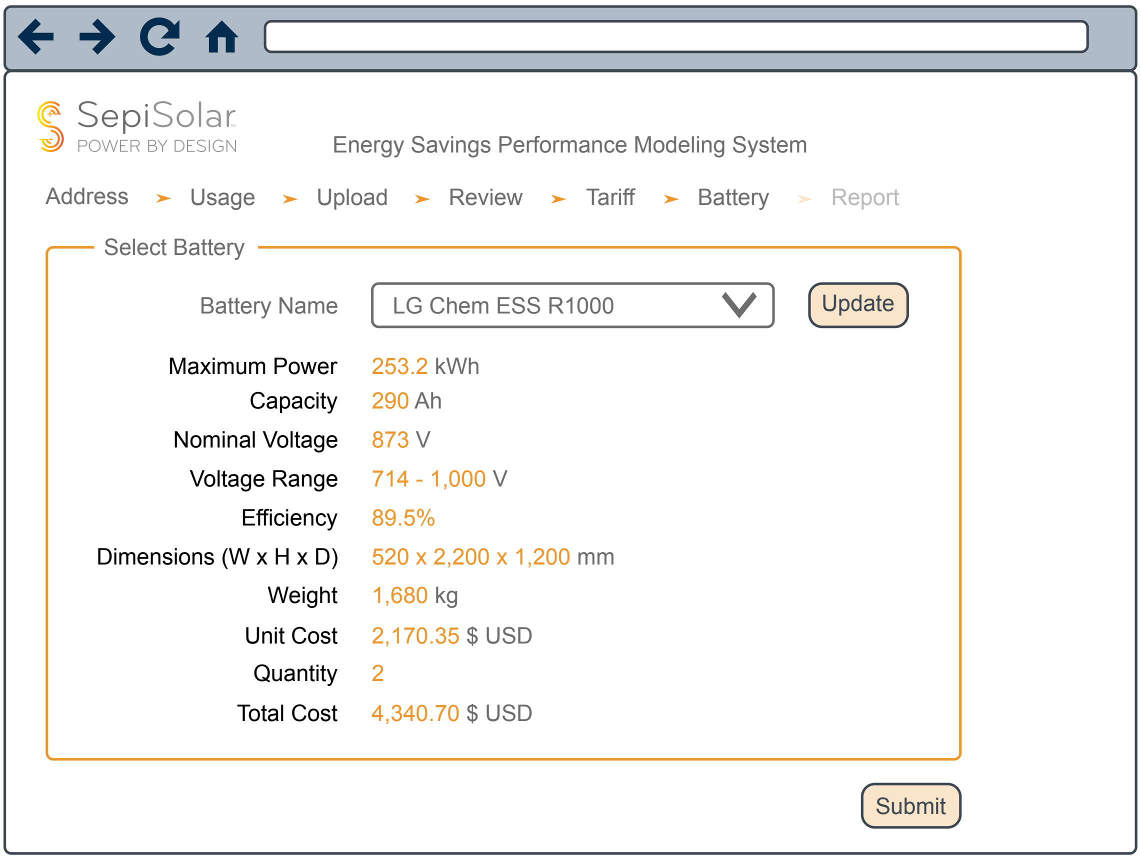
Utility Company	California Hydro
Tariff Name	Commercial Tariff
Qualifier	Option A

Figure 5.7 Wizard Page – Tariff

If the utility company has already been provided (due to earlier requirements), then it may be possible to preselect that company on this page. Changing the utility company name cascades the available options under the tariff name. Similarly, selecting a tariff name subsequently constrains the number of qualifiers listed in the corresponding drop-down.

5.3.6 Battery

Users can select the type of battery to use for energy storage. Changing the battery name updates the page with the selected battery's electrical, physical, performance, and monetary properties.


SepiSolar
POWER BY DESIGN
Energy Savings Performance Modeling System
Address ➤ Usage ➤ Upload ➤ Review ➤ Tariff ➤ Battery ➤ Report
Select Battery
Update
Submit

Battery Name	LG Chem ESS R1000
Maximum Power	253.2 kWh
Capacity	290 Ah
Nominal Voltage	873 V
Voltage Range	714 - 1,000 V
Efficiency	89.5%
Dimensions (W x H x D)	520 x 2,200 x 1,200 mm
Weight	1,680 kg
Unit Cost	2,170.35 $ USD
Quantity	2
Total Cost	4,340.70 $ USD

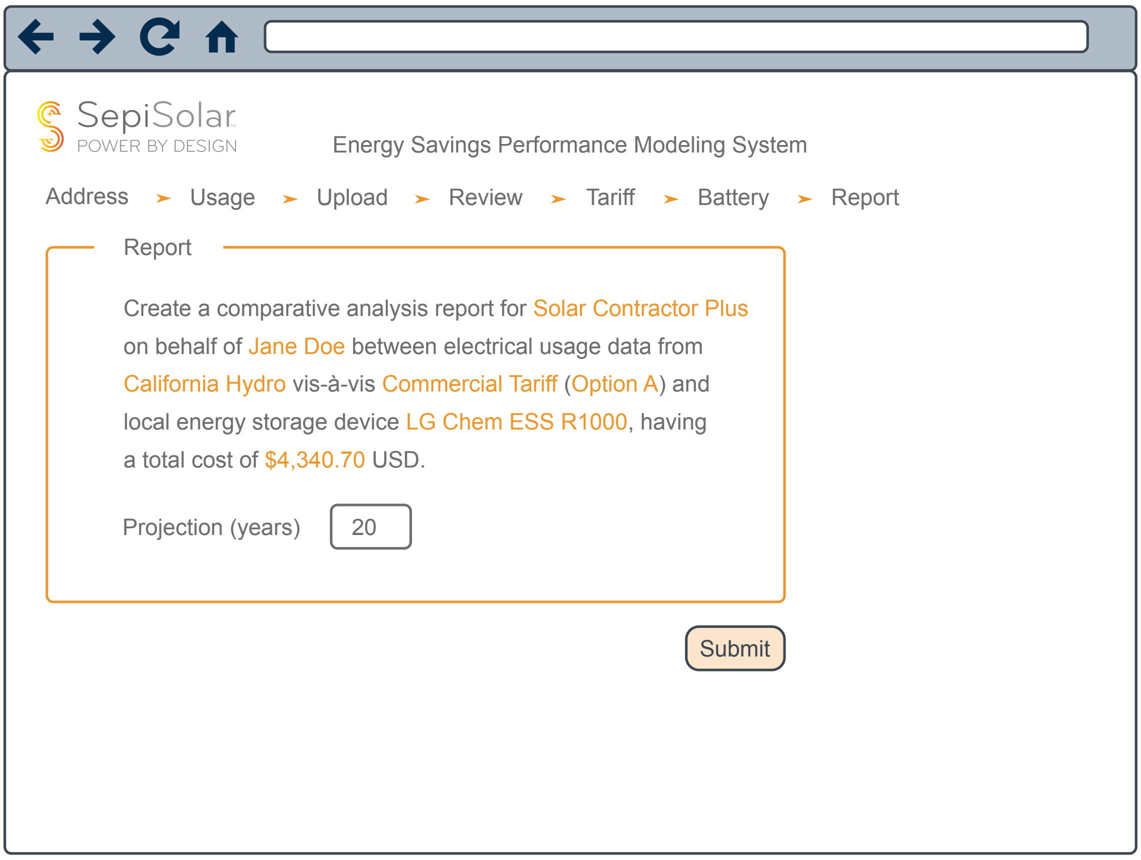
Figure 5.8 Wizard Page – Battery

Although not shown, users may have the option of filtering the list of available batteries. Possible filters include:

- Unit cost
- Total cost
- Brand name
- Dimensions

5.3.7 Report

For the first release it was decided that static output, rather than dynamic charts, would be acceptable. The report page provides a summary of what will be included in the ROI report. Users may set the projection value to change how far into the future the system will calculate the crossover point for cash flow returns.


SepiSolar POWER BY DESIGN

Energy Savings Performance Modeling System

Address ➤ Usage ➤ Upload ➤ Review ➤ Tariff ➤ Battery ➤ Report

Report

Create a comparative analysis report for Solar Contractor Plus on behalf of Jane Doe between electrical usage data from California Hydro vis-à-vis Commercial Tariff (Option A) and local energy storage device LG Chem ESS R1000, having a total cost of $4,340.70 USD.

Projection (years) 20

Figure 5.9 Wizard Page – Report

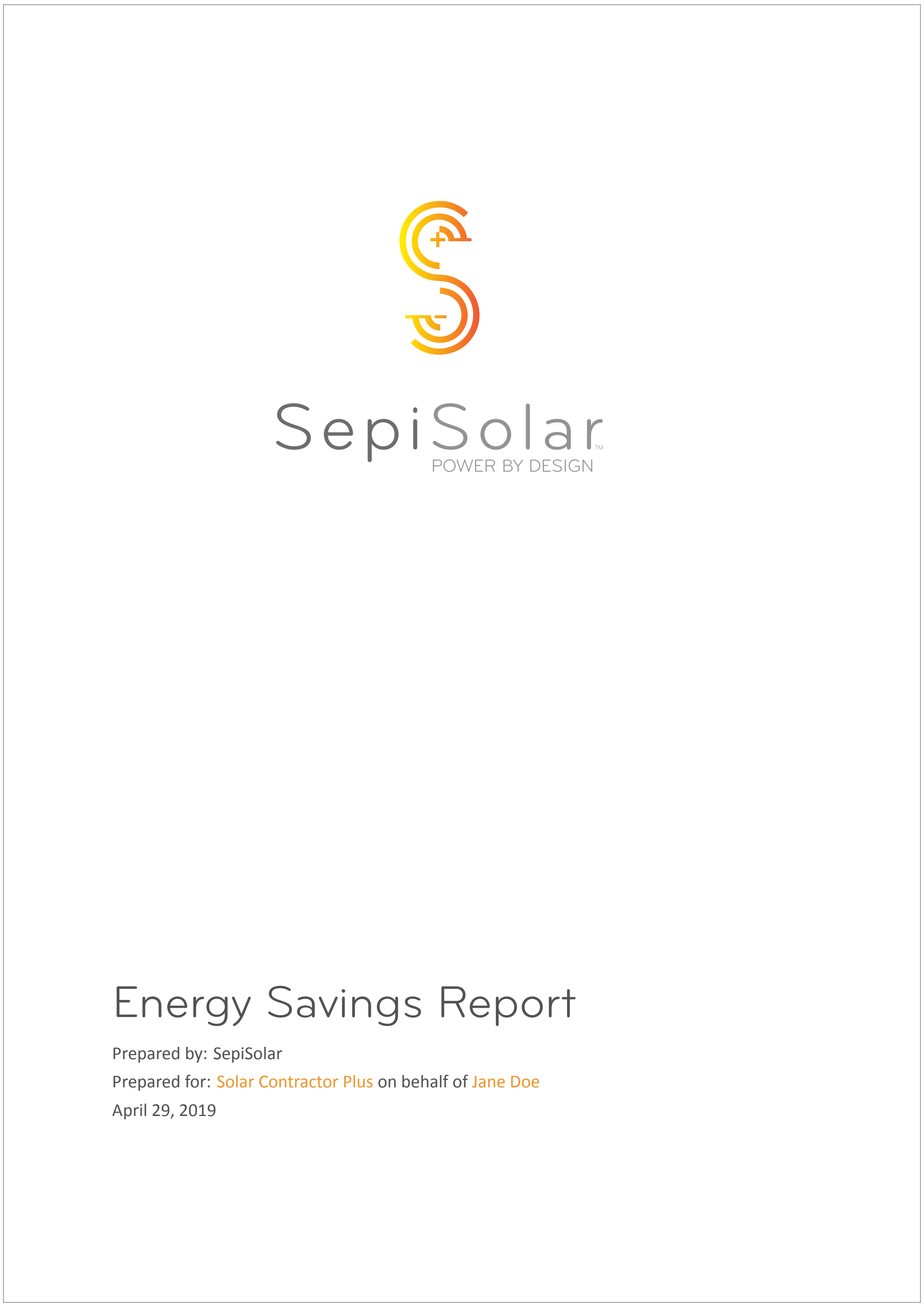
The output from the report is shown in the next section.

5.4 Output

The purpose of the system is to determine whether a purchaser is a good candidate for energy storage. Generated reports must include the following information:

- **Current costs** – A table listing 12 months of electrical costs and total amount using the utility company's tariffs and purchaser's electricity usage data.
- **Projected costs** – A table listing 12 months of projected costs and total amount using the selected ESS and purchaser's electricity usage data.
- **Return costs** – An estimated cash flow analysis to determine the ROI cross-over.

A sample report is shown on the following pages.



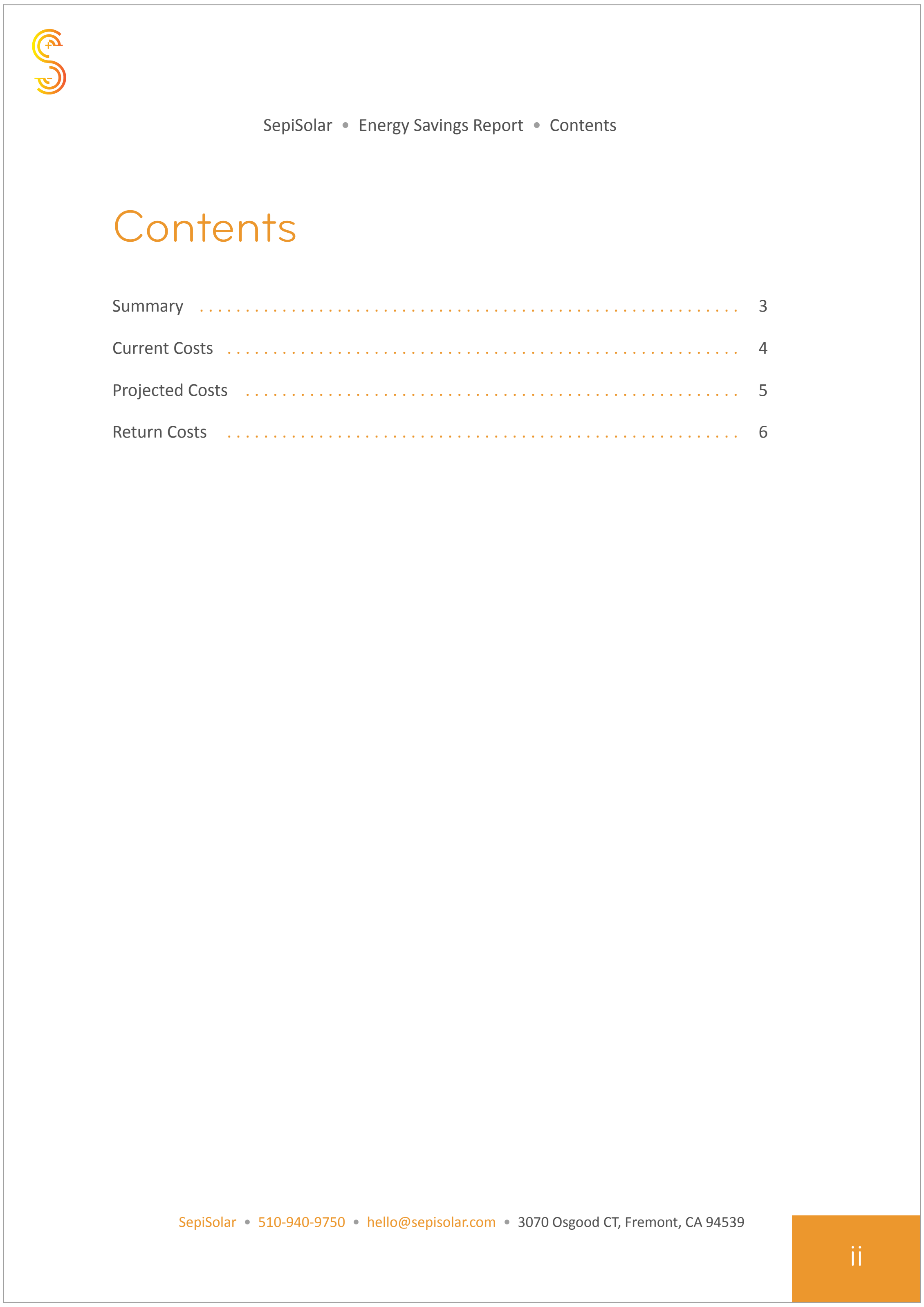
SepiSolar™
POWER BY DESIGN

Energy Savings Report

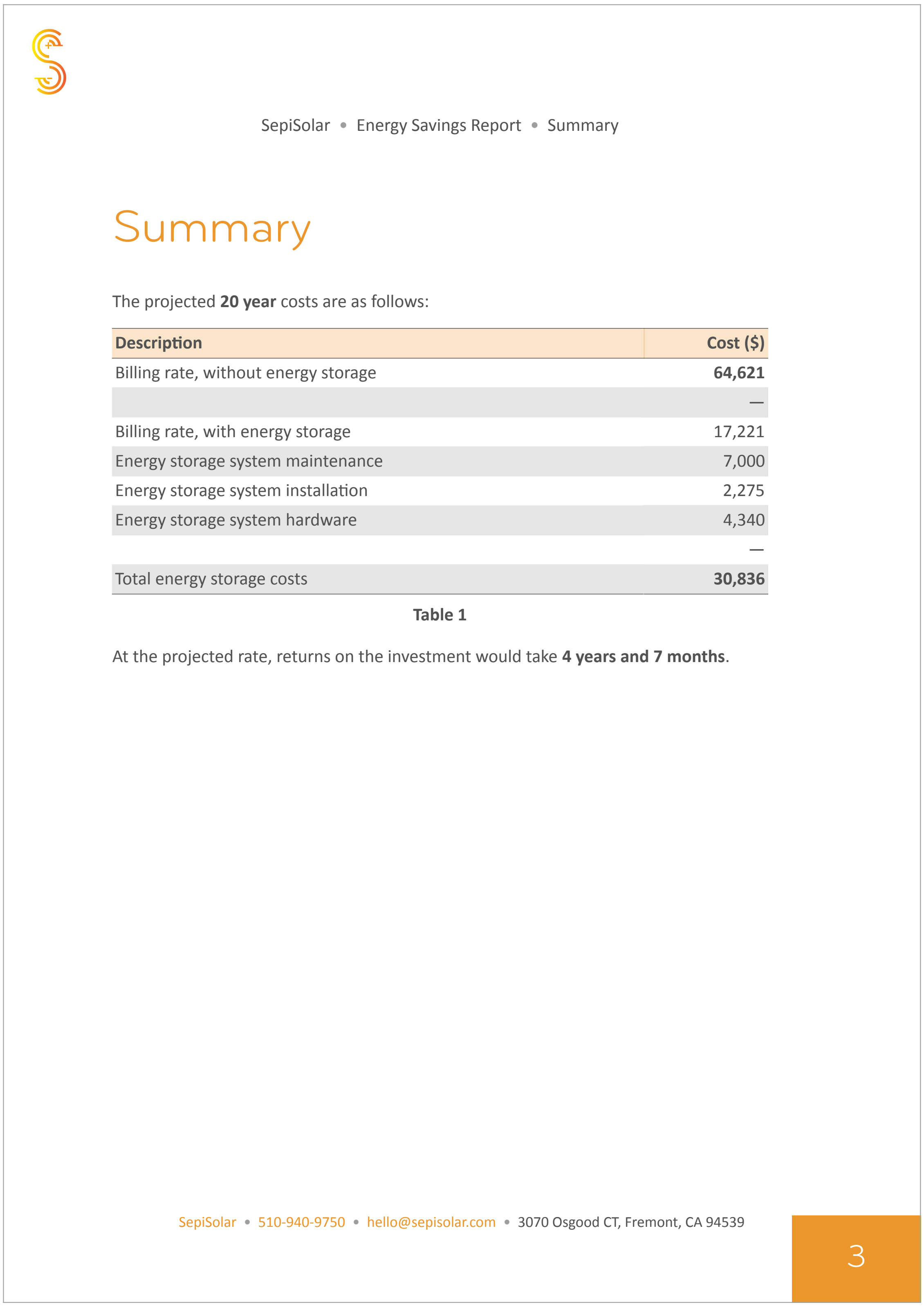
Prepared by: SepiSolar

Prepared for: Solar Contractor Plus on behalf of Jane Doe

April 29, 2019



Contents



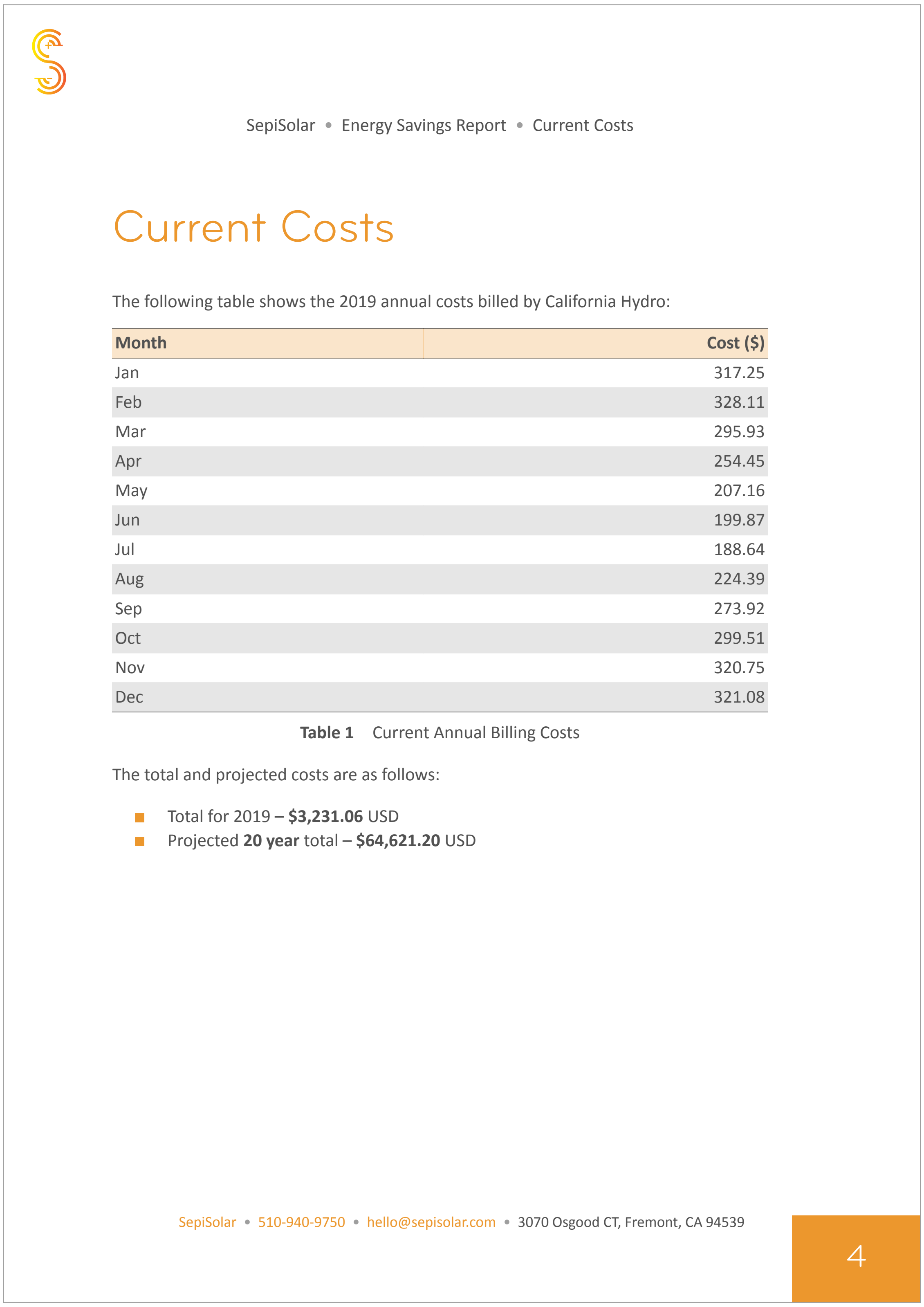
Summary

The projected **20 year** costs are as follows:

Description	Cost ($)
Billing rate, without energy storage	**64,621**
	—
Billing rate, with energy storage	17,221
Energy storage system maintenance	7,000
Energy storage system installation	2,275
Energy storage system hardware	4,340
	—
Total energy storage costs	**30,836**

Table 1

At the projected rate, returns on the investment would take **4 years and 7 months**.



Current Costs

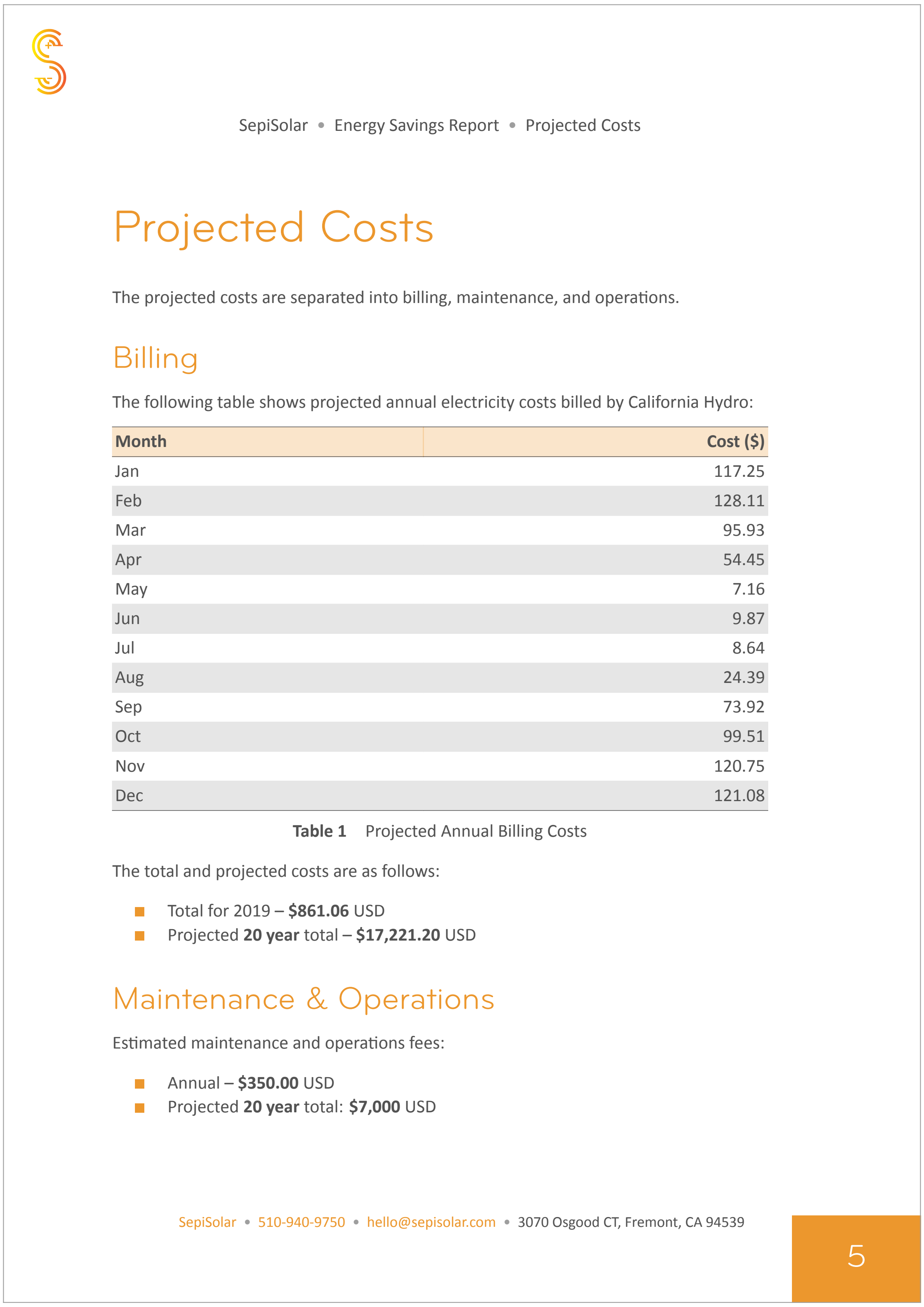
The following table shows the 2019 annual costs billed by California Hydro:

Month	Cost ($)
Jan	317.25
Feb	328.11
Mar	295.93
Apr	254.45
May	207.16
Jun	199.87
Jul	188.64
Aug	224.39
Sep	273.92
Oct	299.51
Nov	320.75
Dec	321.08

Table 1 Current Annual Billing Costs

The total and projected costs are as follows:

- Total for 2019 – **$3,231.06** USD
- Projected **20 year** total – **$64,621.20** USD



Projected Costs

The projected costs are separated into billing, maintenance, and operations.

Billing

The following table shows projected annual electricity costs billed by California Hydro:

Month	Cost ($)
Jan	117.25
Feb	128.11
Mar	95.93
Apr	54.45
May	7.16
Jun	9.87
Jul	8.64
Aug	24.39
Sep	73.92
Oct	99.51
Nov	120.75
Dec	121.08

Table 1 Projected Annual Billing Costs

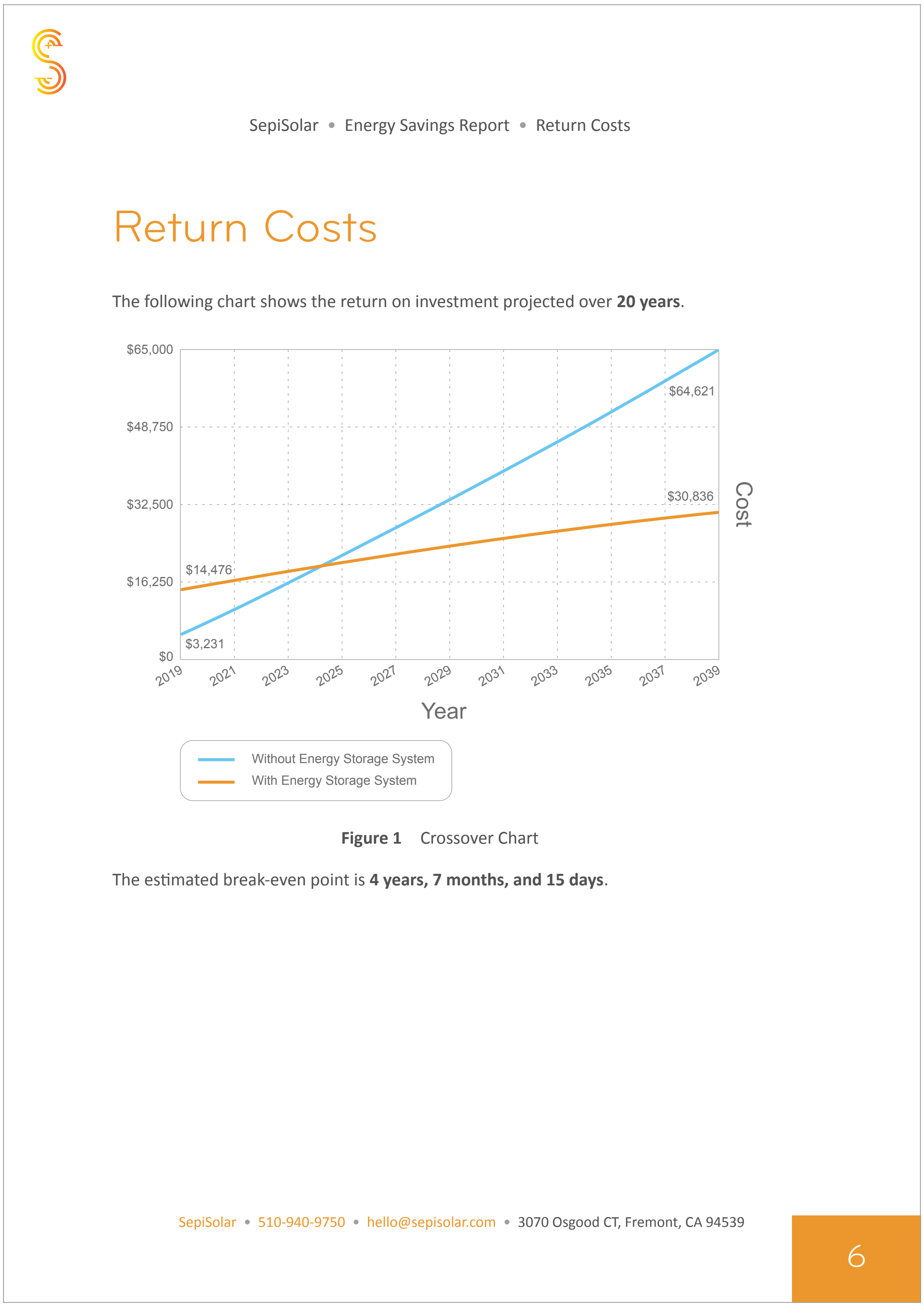
The total and projected costs are as follows:

- Total for 2019 – **$861.06** USD
- Projected **20 year** total – **$17,221.20** USD

Maintenance & Operations

Estimated maintenance and operations fees:

- Annual – **$350.00** USD
- Projected **20 year** total: **$7,000** USD



Return Costs

The following chart shows the return on investment projected over **20 years**.

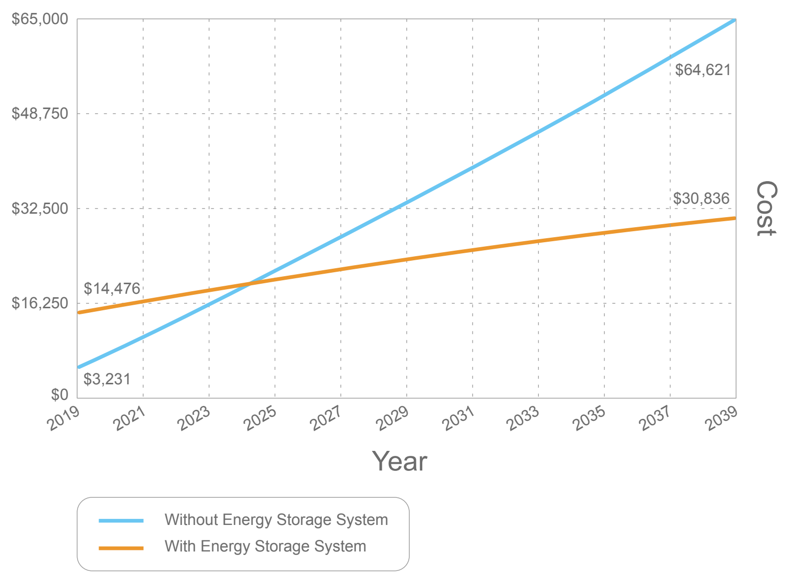


Figure 1 Crossover Chart

The estimated break-even point is **4 years, 7 months, and 15 days**.

6 Architectural Strategies

This section describes design decisions and trade-offs that affect the overall system.

6.1 Interactivity

To simplify development of the first release, a non-interactive report is produced. While this limits the amount of real-time tweaking users can perform, decreasing the scope will shorten the project timeline.

6.2 User Interface Paradigms

The application is first developed without any JavaScript dependencies. Once the application is working using basic HTML forms, JavaScript may be used to enhance functionality.

6.3 Future Enhancements

After a stable version is released, subsequent enhancements can include interactive reports and additional search criteria (for batteries and PV systems).

6.4 External Databases

To protect PII, the production **Purchasers** database will be locked behind a firewall. The only server allowed to use the database will be the web application server and the database administrator's computer.

6.5 Programming Languages

This section describes possible languages and frameworks to use for the application.

6.5.1 Java

Java is a ubiquitous programming language that offers numerous web application development frameworks and modern facilities such as lambda expressions. The Vert.x framework is one of the fastest web development platforms available; the framework includes numerous integrations for security, templating, transport protocols, and routing.

6.5.2 R

The R programming language contains numerous packages for statistical computing and technical documentation, including:

- optimx package, which provides checks for Kuhn-Karush-Tucker (KKT) conditions
- superheat package, which can generate heatmaps
- knitr package, which is a general-purpose dynamic report generation tool

R can be integrated in various ways:

- PL/R project, which integrates with PostgreSQL
- Renjin, a JVM-based R interpreter for Java

6.5.3 SQL

The application uses highly relational data, thus ANSI SQL (rather than NoSQL) is sufficient for database transactions.

6.6 Report Generation

ConTEXt is a TEX-based typesetting engine that can produce high-quality PDF documentation. Using pandoc and ConTEXt provides a powerful toolchain for producing documentation from Markdown. Further integration with knitr provides the ability to integrate R code with Markdown, thereby creating dynamic documents.

The Java-based FlyingSaucer library can create dynamic documents using FreeMarker templates. Leveraging HTML and CSS makes FlyingSaucer easier to learn than ConTEXt, at the expense of far fewer typesetting features.

The Java-based JasperReports library, along with the Jaspersoft Studio IDE, provides the ability to produce high-quality, pixel-perfect reports. JasperReports excels at creating tabular reports; however, it is not a typesetting engine.

6.7 Operating System

An Arch Linux distribution, such as Antergos, provides performance and a wealth of POSIX-compliant tools. While Antergos is suitable as a development platform, its rolling updates may negatively impact stability. Given that the application will likely be deployed to the cloud, Ubuntu would provide the stability necessary for a production environment. Futhermore, an AWS-tuned kernel offers performance benefits.

7 Policies

Having policies in place prior to development can help improve code quality and increase the project's chances of success. Such policies are beyond the scope of this document, but include:

- Requirements traceability
- Coding guidelines and conventions
- Software maintenance planning
- Issue tracking and problem resolution
- Development operations infrastructure
- Building and continuous integration
- Application of continuous improvements

Each of these policies warrants a separate design document.

8 Data Design

This data design describes a subset of entities for the purchasers' relational database.

8.1 Data Dictionary

The data dictionary uses the following terms:

- **PK** – Primary key (surrogate)
- **FK** – Foreign key

All columns are constrained as `NOT NULL`, without exception.

Valid values for common data types can be enforced using checked constraints and are defined as follows:

Data Type	Valid Values
`BIGINT`	`0` to `9,223,372,036,854,775,807`, inclusive
`INTEGER`	`0` to `2,147,483,647`, inclusive

Table 8.1 Common Data Types

8.1.1 Electricity Usage

The electricity usage entity captures electricity data usage by a purchaser and is defined as follows:

Name	Description	Data Type	Valid Values
ID	PK	`BIGINT`	
PURCHASER_ID	FK to purchaser	`INTEGER`	
MEASURED	Usage interval started	`TIMESTAMP`	Seconds since epoch
CONSUMPTION	Power used (kWh)	`NUMERIC(10,3)`	

Table 8.2 Electricity Usage

To save space and eliminate some data conversion, timezones must be derived from the purchaser's utility company timezone.

8.1.2 Utility Company

The utility company entity captures distinct utility companies by name and is defined as follows:

Name	Description	Data Type	Valid Values
ID	PK	`BIGINT`	
LABEL	Company name	`VARCHAR2(255)`	UTF-8 string

Table 8.3 Utility Company

8.1.3 Utility Company Address

The utility company address entity relates a utility company to an address and is defined as follows:

Name	Description	Data Type	Valid Values
ID	PK	`BIGINT`	
UTILITY_COMPANY_ID	FK to utility company	`BIGINT`	
ADDRESS_ID	FK to address	`BIGINT`	

Table 8.4 Utility Company Address

8.1.4 Address

The address entity relates parts of an address and is defined as follows:

Name	Description	Data Type	Valid Values
ID	PK	`BIGINT`	
STREET_ID	FK to street	`BIGINT`	
CITY_REGION_ID	FK to city region	`BIGINT`	
ZIPCODE_ID	FK to zipcode	`BIGINT`	
COUNTRY_ID	FK to country	`BIGINT`	

Table 8.5 Address

The address can be part of the Party Model.

8.1.5 City Region

The city region entity relates a city with a region and is defined as follows:

Name	Description	Data Type	Valid Values
ID	PK	`BIGINT`	
CITY_ID	FK to city	`BIGINT`	
REGION_ID	FK to region	`BIGINT`	
TIMEZONE_ID	FK to timezone	`BIGINT`	

Table 8.6 City Region

8.1.6 Purchaser Utility Company

The purchaser utility company entity relates a purchaser with a particular utility company and is defined as follows:

Name	Description	Data Type	Valid Values
ID	PK	`BIGINT`	
PURCHASER_ID	FK to purchaser	`INTEGER`	
UTILITY_COMPANY_ID	FK to utility company	`BIGINT`	

Table 8.7 Purchaser Utility Company

By joining the utility company with its address, the timezone for a purchaser's electricity usage data can be obtained.

8.2 Entity-Relationship Diagrams

The purpose of these entity-relationship diagrams (ERD) is to present the reader with extracts of the data model. For brevity, only one fairly trivial ERD is included.

8.2.1 Electricity Usage to Timezone Relation

The following diagram depicts how to derive the timezone for a purchaser's electricity usage data point:

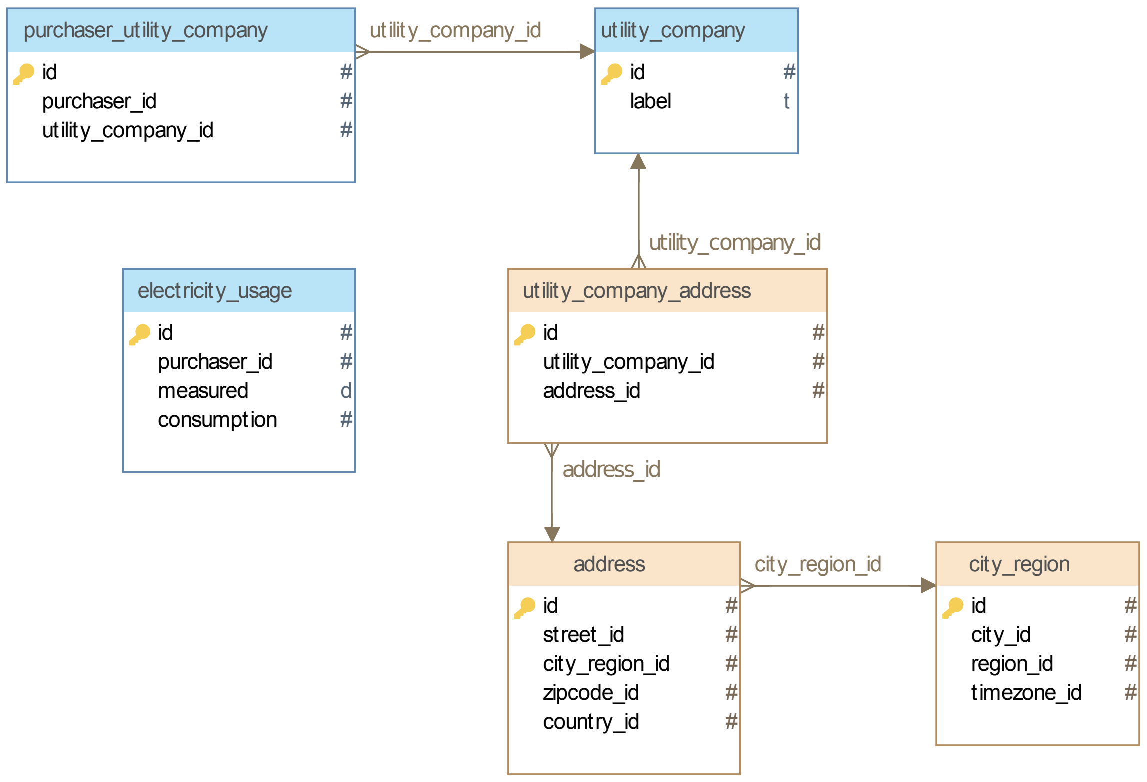


Figure 8.1 Electricity Usage to Timezone Relation

Since the timezone information is not stored in the `electricity_usage` table, which may span multiple billions of rows, if the timezone is needed, it can be retrieved via the purchaser's utility company's timezone.

Some regions have more than one timezone, therefore the city and the region are required to determine the timezone where the measurement was made.

9 Detailed System Design

This section provides a detailed description of the **Modeling System**.

9.1 Classification

The **Modeling System** is a component that provides core functionality required for the proposed web application.

9.2 Definition

The **Modeling System** gathers information from **Users**, provides graphs to help validate volumous data, and generates documentation to assist solar contractors with determining whether a **Purchaser** offers a viable ESS business opportunity.

9.3 Responsibilities

The **Modeling System** has the following responsibilities:

- Provide a web-based front-end for **Users**
- Authenticate **Users** based on pre-existing credentials
- Manage **User** sessions
- Collect and store electricity usage data
- Request ancillary data from other systems, including:
    - Utility company tariff data
    - Battery product data
    - Photovoltaic product data
    - Solar generation data
- Submit electricity usage data requests
- Compute optimum allocation of incoming electricity for battery charging
- Generate heatmap and crossover graphs
- Calculate and tabulate monthly energy costs
- Compile and summarize information into a report

The **Modeling System** does not expose or provide external services.

9.4 Constraints

The **Modeling System** is subject to the following constraints:

- Machine-readable data is given in comma-separated value (CSV) format
- **Users** are managed externally to the application
- PII data is purged after 5 years

9.5 Resources

The following hardware resources are recommended for use by the **Modeling System** component:

- **SSD** – 6 TB of database storage space
  - accommodates 5 years of operations
  - includes 1 TB of application file system space
- **CPU** – Intel Core i9-7980XE (18 cores)
- **RAM** – 64 GB

Storage space was estimated as follows:

- ~184 kB of electricity usage data per **Purchaser**
- ~500 potential **Purchasers** per solar contractor per year
- ~7,000 solar contractors in the United States
- $184kB \times 500 \times 7,000 = 644GB$

9.6 Composition

The **Modeling System** is comprised of the following subcomponents:

- **Web application framework** – Handles web browser requests
- **Authentication and authorization framework** – Enforces web application access restrictions
- **Report generation component** – Produces the **Users'** report
- **Data analytics component** – Computes electricity allocations
- **Graphing component** – Generates charts shown in browsers and in reports

- **Database abstraction layer** – Handles persistence, transactions, connection pools
- **HTTP client component** – Issues REST API requests to other systems

9.7 Libraries

The following table lists recommended software for subcomponents:

Library	Usage
Vert.x	Web application framework
Vert.x Auth	Authentication and authorization
ConTEXt	Typesetting engine for report generation
R	Data analytics component
R	Graphing component
Hibernate	Database abstraction layer
Unirest	HTTP client component

Table 9.1 Subcomponent Software Packages

9.8 Processing

The general steps for calculating the optimal electricity are within the scope of this document; however, more discussion is necessary to formalize the precise processing steps. Formalized processing steps include a description of the following:

- Relevant time or space complexity
- Potential concurrency issues
- Application state changes
- Exception flows
- Compute optimal cost for charging batteries
- Compute ROI crossover

9.9 Algorithms

This section offers algorithms, equations, and formulae that may help determine optimal solutions to computing the system ROI.

9.9.1 Karush–Kuhn–Tucker Conditions

Karush–Kuhn–Tucker (KKT) conditions can help determine minimal and maximal solutions to non-linear problems. Suppose that a function f is given by:

$$f : \mathbb{R}^n \to \mathbb{R}$$

having constraint functions g_i and h_j that are defined by:

$$g_i : \mathbb{R}^n \to \mathbb{R}$$
$$h_j : \mathbb{R}^n \to \mathbb{R}$$

and are continuously differentiable at the point x^*. If x^* is a local optimum and the problem satisifies a given regularity condition, then there exist KKT multipliers for the constants $\mu_i (i = 1, ..., m)$ and $\lambda_j (j = 1, ..., \ell)$ such that stationary values are maximized according to:

$$f(x) : \nabla f(x^*) = \sum_{i=1}^{m} \mu_i \nabla g_i(x^*) + \sum_{j=1}^{\ell} \lambda_j \nabla h_j(x^*)$$

and minimized by $-\nabla f(x^*)$.

9.10 Implementation

The implementation languages include:

Bash	Command language for writing automation scripts
ConTEXt	A document typesetting language
Java	A cross-platform application development language
JavaScript	To enhance, not supplement, front-end web application features
PL/R	An R integration with PostgreSQL's stored procedure language
PL/SQL	A stored procedure language
R	A statistical analysis tool, capable of producing heat maps
SQL	A query language for flat, relational data

10 Testing

Testing the application provides assurances that the system is robust and reliable. This section describes various aspects of testing to help achieve these goals.

10.1 Acceptance Testing

User acceptance testing (UAT) involves manual testing, including penetration testing, of the application to ensure that it conforms to specification. This entails creation of scripts that people can follow that lead to 100% coverage of system functionality. Additionally, UAT includes verifying that the users' needs are met by the system.

10.2 System Testing

End-to-end system testing will be autmomated whenever possible. This includes:

- Verifying third-party API calls
- Database integrity validation
- Network bandwidth and latency checks

10.3 Integration Testing

User interface elements will have 100% coverage for all happy path scenarios.

10.4 Unit Testing

Each low-level component must have at least 75% coverage of its code branches to pass code review.

10.5 Code Quality

Source code must pass inspection by static source code analysis tools. For example. Java tools include:

- FindBugz
- Copy Paste Detector (CPD)
- SonarQube

11 Constraints

This chapter describes non-functional requirements that constrain the system design and implementation.

11.1 Success Metrics

The project will be deemed successful if:

- solar contractors close more deals; and
- ROI calculation times are reduced by an order of magnitude.

This implies that these metrics must be measured before the project is launched.

11.2 Performance

In general, the system shall have the following performance characteristics:

- **Queries** – Database queries must return within 100 milliseconds.
- **Static pages** – Static web pages must return within 1 second.
- **Dynamic pages** – Dynamic web pages must return within 3 seconds.
- **Online reports** – Real-time reports must return within 3 seconds.
- **Offline reports** – Reports generated offline must complete within 5 minutes.

If online reports cannot be generated within the allotted timeframe, it is permissible to generate the report offline for later retrieval.

11.3 Security

The application contains private information.

- Internal and external systems communications must use encrypted protocols.
- Email addresses must be encrypted.
- Passwords must be hashed using a cryptographically secure algorithm and salt.

11.3.1 Database

Databases are configured as follows:

- Firewalled with access limited to whitelisted IP addresses.
- Schemas to separate confidential information.
- Specific roles granted to generic users, such as:
  - **APP_RO** – Application-level, read-only
  - **APP_RW** – Application-level, read-write
  - **DEV_RO** – Developer, read-only
  - **DEV_RW** – Developer, read-write
- Non-production databases must scramble personally identifiable information.

11.4 Licensing

Unless otherwise specified or in the public domain, only third-party software having the following licenses may be used for development purposes:

- AGPL 3
- Apache 2
- BSD 3
- CC0
- GNU GPL 3
- GNU LGPL 3
- MIT
- PostgreSQL
- SIL
- W3C

11.5 Language

All human-readable documentation, including but not limited to internal source code comments and public system API documentation, shall be written using American English.