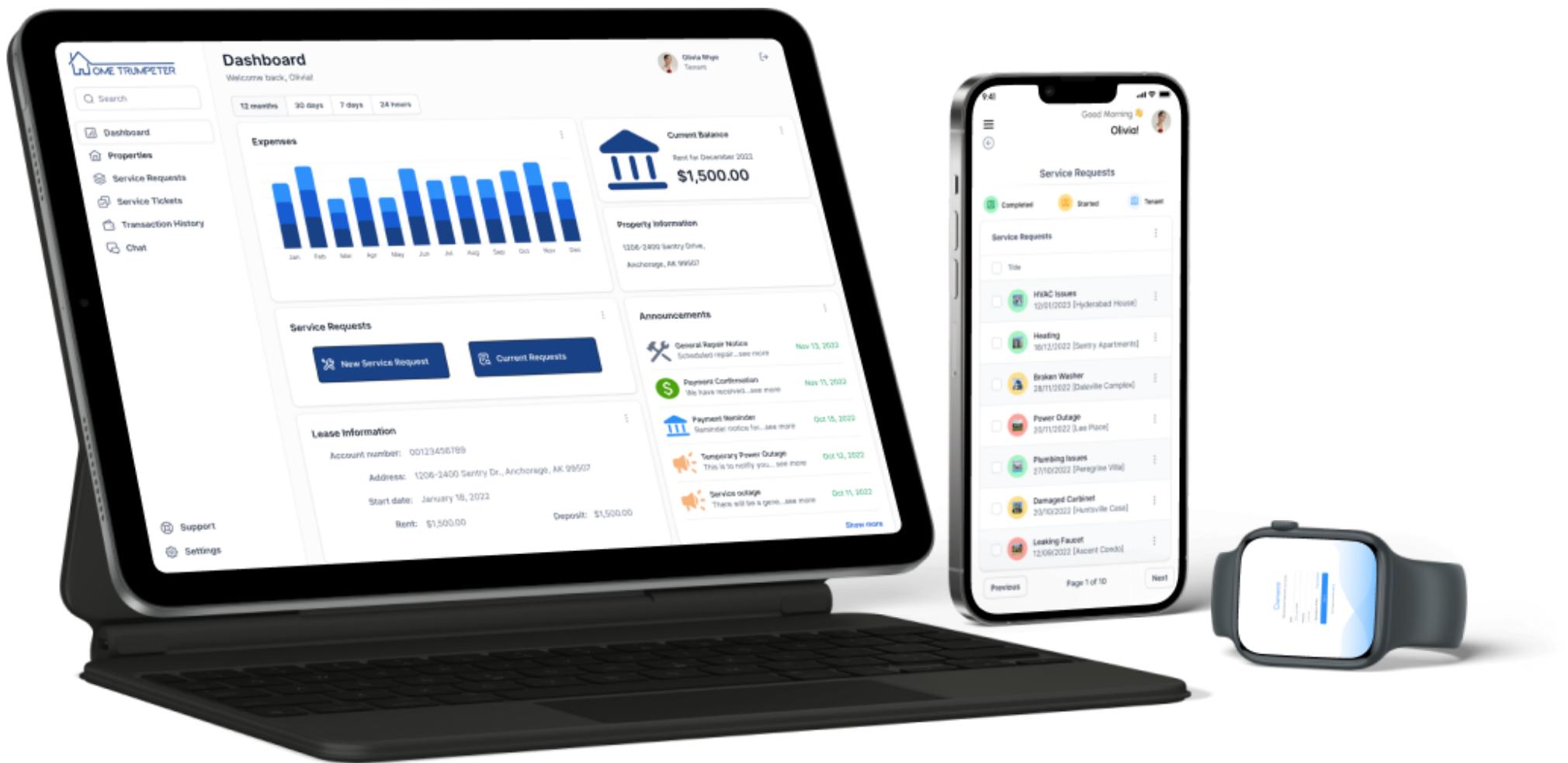


# Prove IT

---



# HomeTrumpeter



## Tenant

Consult and schedule repairs and maintenance using one of the verified service experts in the blink of an eye.

## Home Owner

Easily liaise between tenants and service providers and automate payment collection systems for rent and other property management-related costs.

## Service Provider

Issue and track the progress of property maintenance requests from tenants and home owners.

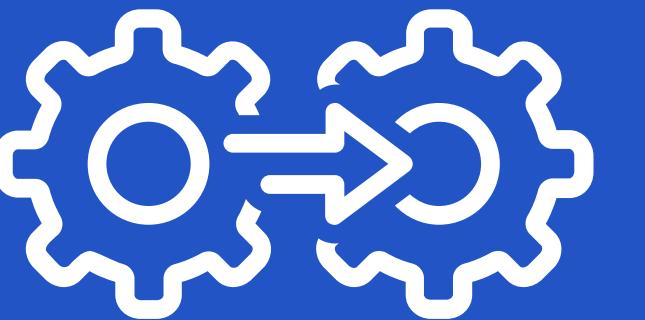
# Problems

It is very difficult to see how the API will work in the real world.

Security



Integration



Feedback



# Solutions



The solutions provided should uphold the mission and vision of HomeTrumpeter LLC. In pursuit of API integration and testing, various factors will be taken into consideration that may affect the business model.

## Integration

The solutions should test the HomeTrumpeter API by using the most **up-to-date technologies** and **frameworks**.

## Security

The solutions will ensure security by conducting **comprehensive tests** against the '**black box**' API.

## Scalability

The solutions should leverage the shift towards **cloud computing** and **re-usability** of complex software components

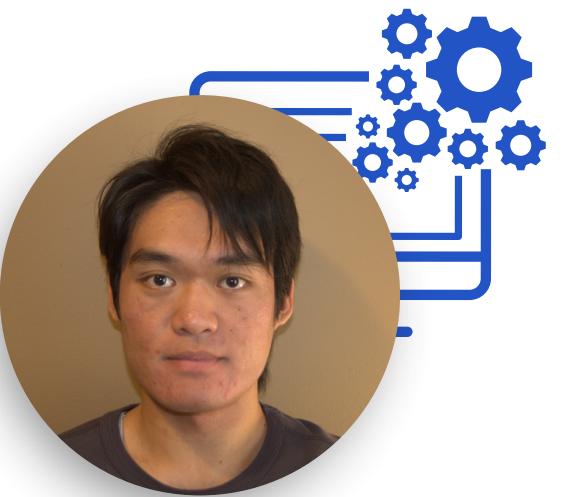
## End User

The solutions offered need to be based on **sound market decisions** so that they can have **impact**.



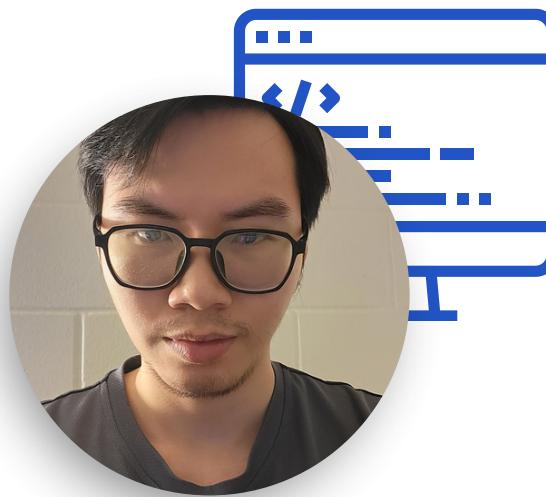
**Liam Kristjanson**

*Scrum Master, Product Owner,  
Project Leader*



**Xiao Zhang**

*Technical Leader*



**Vi Le**

*Lead Developer*



**Arshjot Ghuman**

*Lead Quality Assurance Engineer*



**Usmaan Sahar**

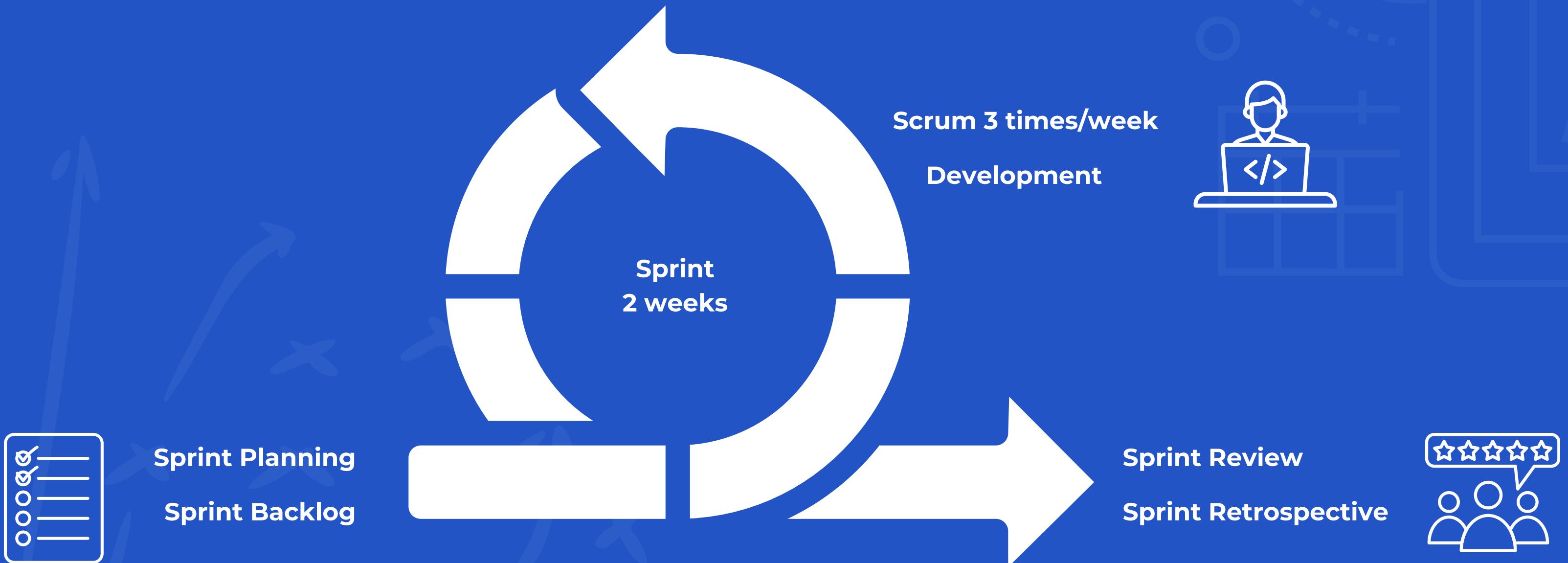
*Lead Systems Analyst / Designer*



**W A Shadman**

*Lead DevSecOps Engineer*

# Agile Release Plan

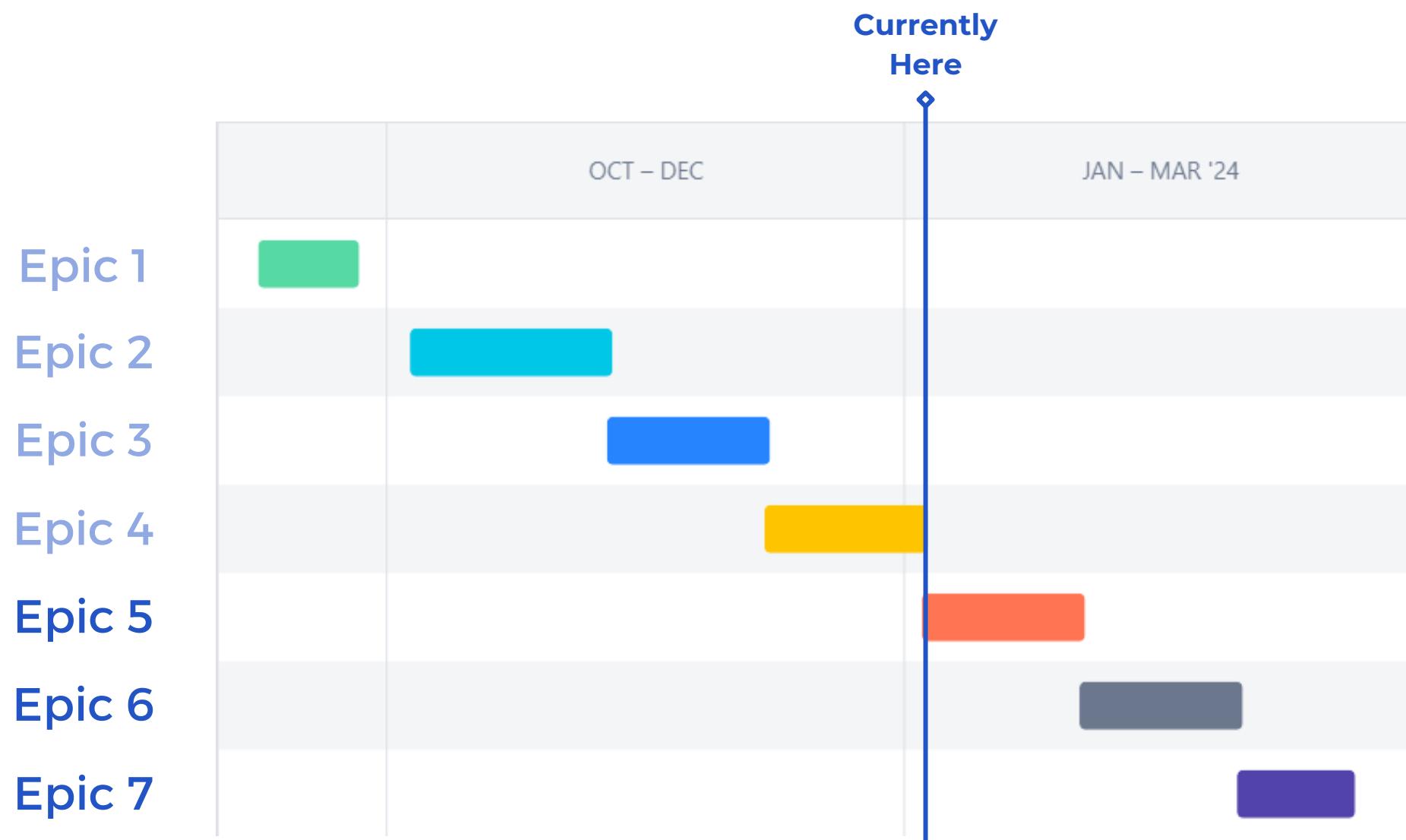


# High-Level Requirements

- 1. Login Portal Using the API Gateway**
- 2. Tenant Onboarding**
- 3. Service Provider Onboarding**
- 4. Service Request Creation**
- 5. Background Check and Notification**
- 6. Service Provider Behavior Prediction Model**



# Timeline



Tentative timeline. More sprints will be added/updated based on stakeholder feedback, as development continues.

2023  
Q3

2023  
Q4

2024  
Q1

## Project Proposal

## Infrastructure

CI/CD Pipeline

Login and Account Creation

Tenant Onboarding

## Services

Service Provider Onboarding

- Dashboard
- Invite system
- Add/remove/edit services

Service Provider Background Check

- Public service provider 'market'
- Transaction process

Service Request Creation

- Homeowner request
- Tenant request
- Confirmation of completion

# Product Backlog

Task Number	Task Name	Resource	Duration	Story Points	Start	Finish	Status
1.0	<b>User Stories</b>			11			
1.1	Identify Key Stakeholders	Project Team		2	06 Sep 2023 1:00PM	06 Sep 2023 2:15PM	Done
1.2	Form Project Team	Project Manager		1	06 Sep 2023 1:00PM	06 Sep 2023 2:15PM	Done
1.3	Project Proposal	Project Team		8	06 Sep 2023 1:00PM	25 Sep 2023 11:59PM	Done
2.0	<b>Product Backlog</b>			4			
2.1	Create Product Backlog	Product Owner		3	13 Sept 2023 1:00PM	30 Sept 2023 12:00PM	Done
2.2	Assign Story Point Estimations	Product Owner		1	27 Sept 2023 1:00PM	1 Oct 2023 12:00PM	Done
3.0	<b>High Level Sprint Planning</b>			3			
3.1	Create Project Plan	Project Manager		3	27 Sept 2023 1:00PM	06 Oct 2023 11:59PM	Done
3.2	Approve Project Plan	Stakeholders		0			Done
4.0	<b>Sprint - 1 - Infrastructure and Learning</b>			4			
4.1	Sprint 1 - Planning Meeting	ALL		0	05 Oct 2023 6:00PM	05 Oct 2023 7:00PM	Done
4.3	Login Gateway Wireframe	Usmaan		2	05 Oct 2023 7:00PM	12 Oct 2023 10:00AM	Done
4.4	Add Property Wireframe	Usmaan		2	05 Oct 2023 7:00PM	12 Oct 2023 10:00AM	Done
5.0	<b>Sprint - 2 - Login with API Portal I</b>			27			
5.1	Sprint 2 - Planning / Sprint 1 Demo	ALL		0	12 Oct 2023 6:00PM	12 Oct 2023 7:00PM	Done
5.2	Develop CI/CD Pipeline - Simple Deployment	QA Engineer		3	05 Oct 2023 7:00PM	16 Oct 2023 7:30PM	Done
5.3	Homeowner/Manager Create Account	Lead Programmer		8	18 Oct 2023 12:00PM	25 Oct 2023 12:00PM	Done
5.4	Homeowner/Manager Login	Technical Lead		5	12 Oct 2023 6:00PM	18 Oct 2023 12:00PM	Done
5.6	Prototype - Create Account/Login	Product Owner		1	12 Oct 2023 6:00PM	13 Oct 2023 12:00PM	Done
5.7	Prototype - Add Property	Product Owner		1	13 Oct 2023 12:00PM	13 Oct 2023 6:00PM	Done
5.8	Prototype - Invite Tenant	Product Owner		1	13 Oct 2023 6:00PM	16 Oct 2023 6:00PM	Done
5.9	Prototype - Invite Service Provider	Lead Designer		1	12 Oct 2023 6:00PM	21 Oct 2023 12:00PM	Done
5.10	Prototype - Create Service Request	Lead Designer		1	12 Oct 2023 6:00PM	21 Oct 2023 12:00PM	Done
5.11	Research Testing Requirements and Tools	DevSecOps		3	16 Oct 2023 12:00PM	20 Oct 2023 12:00PM	Done
5.12	Research and Select Tech Stack	Technical Lead		3	12 Oct 2023 6:00PM	18 Oct 2023 2:00PM	Done
6.0	<b>Sprint - 3 - Login with API Portal II</b>			37			
6.1	Sprint 3 - Planning / Sprint 2 Demo	ALL		0	26 Oct 2023 6:30PM	26 Oct 2023 7:30PM	Done
6.1.1	Sprint 2 Retrospective	Project Team		0	26 Oct 2023 7:30PM	26 Oct 2023 8:00PM	Done
6.2	Protect Routes	Technical Lead		5	26 Oct 2023 7:30PM	27 Oct 2023 12:00PM	Done
6.3	CI/CD Pipeline - Promotion of Environments	QA Engineer		8	26 Oct 2023 7:30PM	6 Nov 2023 12:00PM	Done
6.4	Implement Unit Testing Framework	QA Engineer		5	26 Oct 2023 7:30PM		Done
6.5	CI/CD Pipeline - Implement Security Scans	QA Engineer/DevSecOps		3	26 Oct 2023 7:30PM	31 Oct 2023 12:00PM	Done
6.6	Homeowner - Dashboard Skeleton	Lead Developer		3	26 Oct 2023 7:30PM	3 Nov 2023 12:00PM	Done
6.8	Identify Abuse Cases	DevSecOps		2	1 Nov 2023 12:00PM	8 Nov 2023 12:00PM	Done
6.9	Threat Modelling	DevSecOps/Lead Design		3	1 Nov 2023 12:00PM	8 Nov 2023 12:00PM	Done
6.11	Homeowner - Verify Phone Number	Project Lead		2	26 Oct 2023 7:30PM	1 Nov 2023 12:30PM	Done
6.11	Select Account Role	Project Lead		1	30 Oct 2023 12:00PM	1 Nov 2023 12:30PM	Done
6.10	Login API - Testing	QA Engineer		5			Done
7.0	<b>Sprint - 4 - Tenant Onboarding I</b>			34			
7.1	Sprint 4 - Planning / Sprint 3 Demo	ALL		0	09 Nov 2023 6:30PM	09 Nov 2023 7:30PM	Done
7.2	Backend Server/API Proxy	Technical Lead		8			Done
7.4	Standardize Styling Rules	Lead Designer		2			Done
7.5	CI/CD Pipeline - Deploy New Backend	QA Engineer/Project Lead		8			Done
7.6	Refactor Project - Divide Frontend/Backend	Lead Developer/Tech Lead		2			Done
7.7	CI/CD Pipeline - Link Frontend with new backend	QA Engineer / Project Lead		3			Done
7.6	Research Secure Backend Server/API Proxy	DevSecOps Engineer		3			Done
7.10	System Study Review	Project Team		8	13 Nov 2023 1:00PM	13 Nov 2023 2:15PM	Done

8.0	<b>Sprint - 5 - Tenant Onboarding II</b>			19			
8.1	Sprint 5 - Planning / Sprint 4 Demo	ALL		0	30 Nov 2023 6:30PM	30 Nov 2023 7:30PM	Done
8.2	Implement Sessions for Authorization	Technical Lead/DevSecOps		8	23 Nov 2023 6:30PM		Done
8.9	Landing Page	Project Manager		3	23 Nov 2023 6:30PM	27 Nov 2023 12:30PM	Done
	Direct user to email on account creation	Unassigned		1	27 Nov 2023 2:00PM		Done
	Fix React Refresh Issue	Project Manager		2	27 Nov 2023 8:00PM	1 Dec 2023 12:00PM	Done
	Homeowner - Add Property Hook/Backend	Lead Developer		5	23 Nov 2023 6:30PM		Done
8.0	<b>Sprint - 5.5 - Tenant Onboarding II</b>			6			
7.8	Homeowner - Add Property React Pages	Lead Designer		3			Done
7.9	Homeowner Dashboard - View Properties	Project Manager		3			Done
9.0	<b>Sprint - 6 - Tenant / Service Provider Onboarding I</b>			44			
9.1	Sprint 6 - Planning / Sprint 5 Demo	ALL		0	07 Dec 2023 6:30PM	07 Dec 2023 7:30PM	Done
9.2	Implement Rate Limiting on Backend	Technical Lead		3	19 Dec 2023 12:00PM	21 Dec 2023 7:30PM	Done
7.7	Implement Standard Testing Practices	QA Engineer		5	10 Dec 2023 12:00PM	12 Dec 2023 7:30PM	Done
8.4	Homeowner Dashboard - View Tenants	Lead Developer		5			Done
	Address Validation	Lead Developer		3	13 Dec 2023 12:00PM	16 Dec 2023 6:00PM	Done
	CI/CD Pipeline - Deploy to Both Data Centres	QA Engineer		3	13 Dec 2023 12:00PM	16 Dec 2023 6:00PM	Done
8.3	Send Tenant Invitation Hook/Backend	Project Manager		8	14 Dec 2023 12:00PM	17 Dec 2023 6:00PM	Done
8.5	Tenant Dashboard	Lead Designer		3			Done
8.7	Tenant Create Account	Project Manager		5	14 Dec 2023 12:00PM	17 Dec 2023 6:00PM	Done
8.8	Tenant Login	Unassigned		3	14 Dec 2023 12:00PM	17 Dec 2023 6:00PM	Done
	Send Tenant Invitation React Page	Project Manager		2	14 Dec 2023 12:00PM	17 Dec 2023 6:00PM	Done
8.6	Tenant Accept Invitation	QA Engineer		3	14 Dec 2023 12:00PM	17 Dec 2023 6:00PM	Done
8.10	Dashboard - Delete Property	Lead Developer		1	18 Dec 2023 12:00PM	19 Dec 2023 6:00PM	Done
10.0	<b>Sprint - 7 - Service Provider Onboarding II</b>			27			
10.1	Sprint 7 - Planning / Sprint 6 Demo	ALL		0	21 Dec 2023 6:30PM	21 Dec 2023 7:30PM	Done
	GROUP LUNCH / Board Games	ALL		0	03 Jan 2023 11:00AM	03 Jan 2023 12:30PM	Done
	Improve Aesthetics of Homeowner Dashboard	Lead Designer		3			Done
8.2	Implement Basic Penetration Testing Plan	DevSecOps/QA Engineer		5	21 Dec 2023 7:30PM	03 Jan 2023 12:30PM	Done
9.2	Invite Service Provider Wireframe	Lead Designer		2			Done
9.3	Send Service Provider Invite	Project Manager		3	27 Dec 2023 9:00AM	28 Dec 2023 12:00PM	Done
9.4	Create Service Provider Account	Project Manager		5	27 Dec 2023 9:00AM	28 Dec 2023 12:00PM	Done
	Validate service provider address	Project Manager		1	29 Dec 2023 9:00AM	29 Dec 2023 9:00PM	Done
	Implement Forgot Password Functionality	Project Manager		2	30 Dec 2023 10:00AM	30 Dec 2023 10:00PM	Done
9.5	Service Provider Login	Project Manager		3	27 Dec 2023 9:00AM	28 Dec 2023 12:00PM	Done
	Basic Service Provider Dashboard	Lead Designer		3	1 Jan 2023 12:00PM	3 Jan 2023 12:00PM	Done

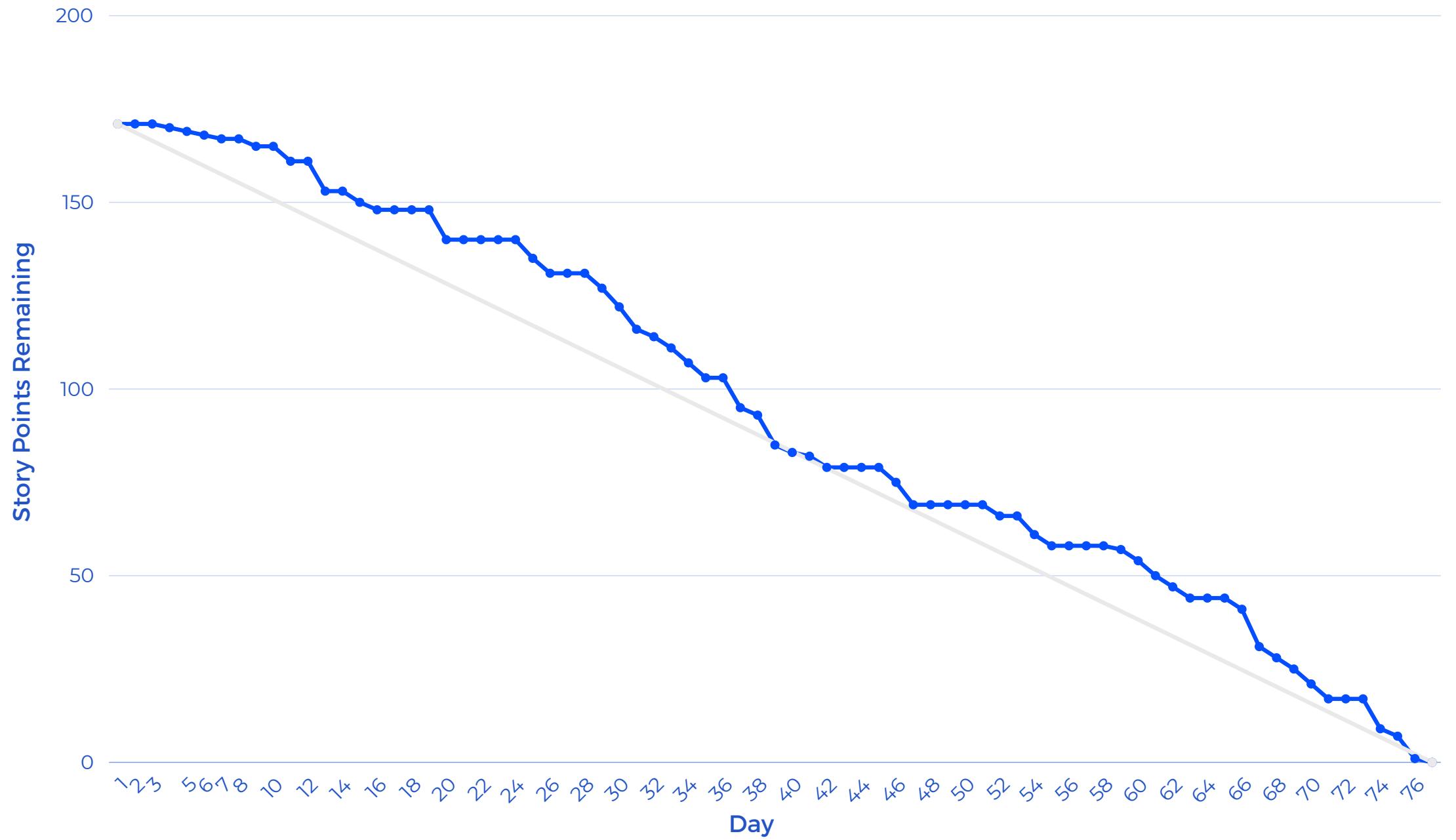
# Product Backlog

Sprint - 8 - Service Request Creation I		42					Done
11.1	Sprint 8 - Planning / Sprint 7 Demo	ALL	0	04 Jan 2024 6:30PM	04 Jan 2024 7:30PM		
10.3	Add Services	Project Manager	3	04 Jan 2024 7:30PM			In Progress
	Homeowner Dashboard - View Tenants	Lead Developer	5	27 Dec 2023 12:00PM			In Progress
	Homeowner Dash - Approve Tenant Questionnaire	Project Manager	5				Not Started
	View Property Details	Lead Developer	3				Not Started
11.4	Homeowner - Request Quote For Service	Unassigned	2				Not Started
11.5	Service Provider Dashboard - View Service Requests	Unassigned	3				Not Started
11.6	SP Dashboard - View Service Request Details	Unassigned	2				Not Started
10.2	Service Provider Dashboard - View My Services	Unassigned	3				Not Started
11.3	Create Service Request - Tenant Initiate	Project Manager	3				In Progress
11.7	Detailed Design Review	ALL	13	15 Jan 2024 1:00PM	15 Jan 2024 2:00PM		Not Started
Sprint - 9 - Service Request Creation II		24					Not Started
12.1	Sprint 9 - Planning / Sprint 8 Demo	ALL	0	18 Jan 2024 6:30PM	18 Jan 2024 7:30PM		
12.2	Send Proposal	Unassigned	2				Not Started
12.3	View Proposed Quotes	Unassigned	3				Not Started
12.4	View Proposed Quote Details	Unassigned	2				Not Started
12.5	Approve Proposed Quote	Unassigned	2				Not Started
12.6	Proposal Notification	Unassigned	2				Not Started
12.7	Proposal Approval Notification	Unassigned	2				Not Started
	Mark Job as In Progress, Completed	Unassigned	3				Not Started
12.8	Service Request - Testing	Unassigned	8				Not Started
Sprint - 10 - Background Check I		26					Not Started
13.1	Sprint 10 - Planning / Sprint 9 Demo	ALL	0	01 Feb 2024 6:30PM	01 Feb 2024 7:30PM		
13.2	Background Check Prototyping	Unassigned	2				Not Started
13.3	Apply for Public Service Provider	Unassigned	3				Not Started
13.4	Send Certn Background Check Request	Unassigned	5				Not Started
13.5	View Status of Background Check	Unassigned	3				Not Started
13.6	Port Homeowner Features to React Native	Unassigned	13				Not Started
Sprint - 11 - Background Check II		16					Not Started
14.1	Sprint 11 - Planning / Sprint 10 Demo	ALL	0	15 Feb 2024 6:30PM	15 Feb 2024 7:30PM		
14.2	Process Completed Background Check from Certn	Unassigned	8				Not Started
14.3	Grant Public Service Provider Status	Unassigned	3				Not Started
14.4	Background Check - Testing	Unassigned	5				Not Started
14.5	Port Tenant Features to React Native	Unassigned	13				Not Started
Sprint - 12 - Service Provider ML Model		19					Not Started
15.1	Sprint 12 - Planning / Sprint 11 Demo	ALL	0	29 Feb 2024 6:30PM	29 Feb 2024 7:30PM		
15.2	Research for ML Model	Unassigned	3				Not Started
15.3	Development of ML Model	Unassigned	8				Not Started
15.4	Integration of ML Model into System	Unassigned	8				Not Started
15.5	Port Service Provider Features to React Native	Unassigned	13				Not Started
Sprint - 13 - Service Provider ML Model		24					Not Started
16.1	Sprint 13 - Planning / Sprint 12 Demo	ALL	0	14 Mar 2024 6:30PM	14 Mar 2024 7:30PM		
16.2	Testing of ML Model	Unassigned	8				Not Started
16.3	Testing of Model Integration	Unassigned	5				Not Started
16.4	System Sign Off	Project Team	3	20-Mar-24	20-Mar-24		Not Started
16.5	Project Completion Seminar	Project Team	8	20-Mar-24	20-Mar-24		Not Started

# Progress

## Sprints 1 to 7

■ Actual ■ Projected



2088 person-hours total

72 person-hours / week



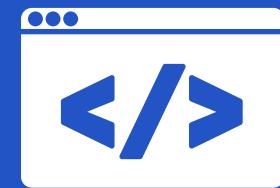
■ Used ■ Available

# Current Sprint

11.0	Sprint - 8 - Service Request Creation I		42				
11.1	Sprint 8 - Planning / Sprint 7 Demo	ALL	0	04 Jan 2024 6:30PM	04 Jan 2024 7:30PM	04 Jan 2024 7:30PM	Done
10.3	Add Services	Project Manager	3	04 Jan 2024 7:30PM			In Progress
	Homeowner Dashboard - View Tenants	Lead Developer	5	27 Dec 2023 12:00PM			In Progress
	Homeowner Dash - Approve Tenant Questionnaire	Project Manager	5				Not Started
	View Property Details	Lead Developer	3				Not Started
11.4	Homeowner - Request Quote For Service	Unassigned	2				Not Started
11.5	Service Provider Dashboard - View Service Requests	Unassigned	3				Not Started
11.6	SP Dashboard - View Service Request Details	Unassigned	2				Not Started
10.2	Service Provider Dashboard - View My Services	Unassigned	3				Not Started
11.3	Create Service Request - Tenant Initiate	Project Manager	3				In Progress
11.7	<b>Detailed Design Review</b>	ALL	13	15 Jan 2024 1:00PM	15 Jan 2024 2:00PM	15 Jan 2024 2:00PM	Not Started

# Architecture

Front-End



Back-End

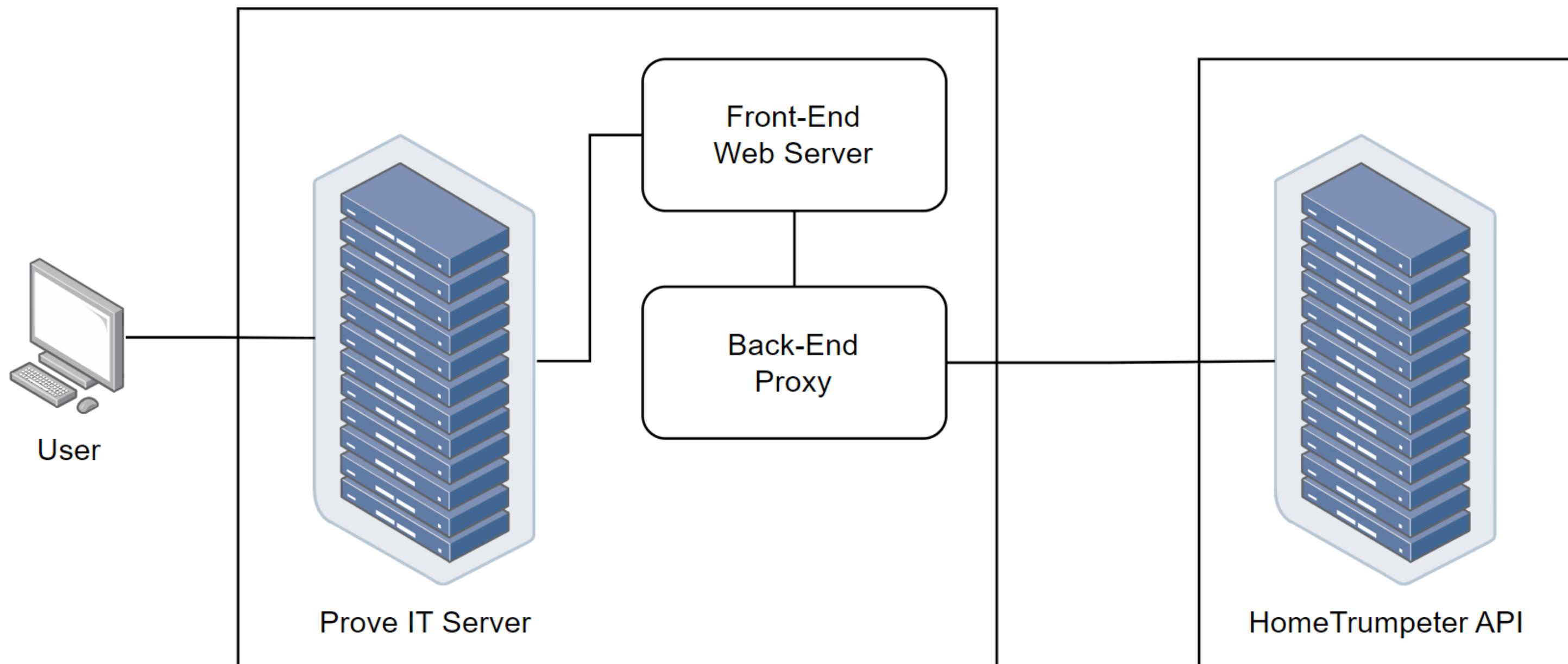


API

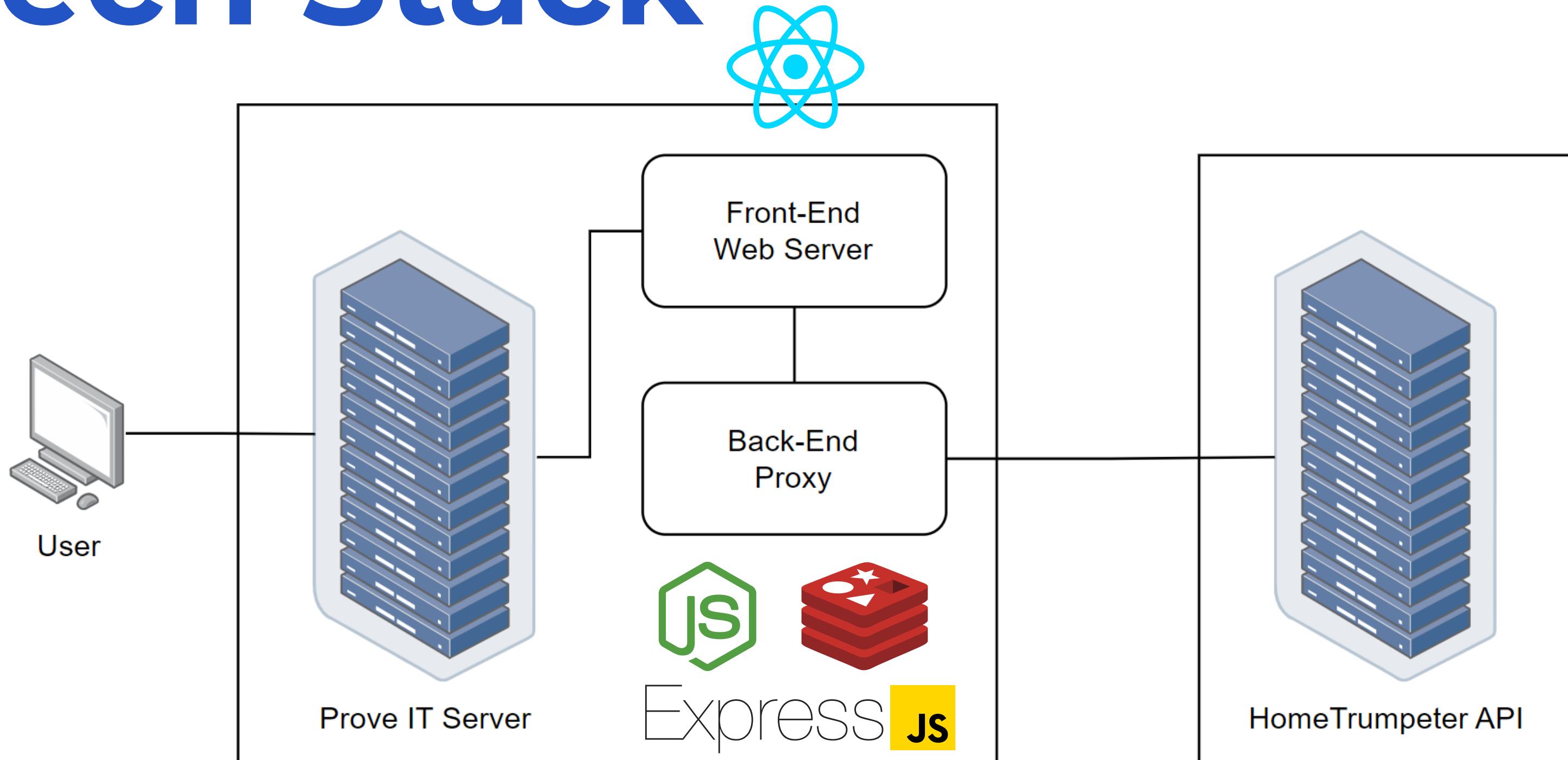


Our system comprised of a full-stack application, utilizing the most popular front-end and back-end frameworks.

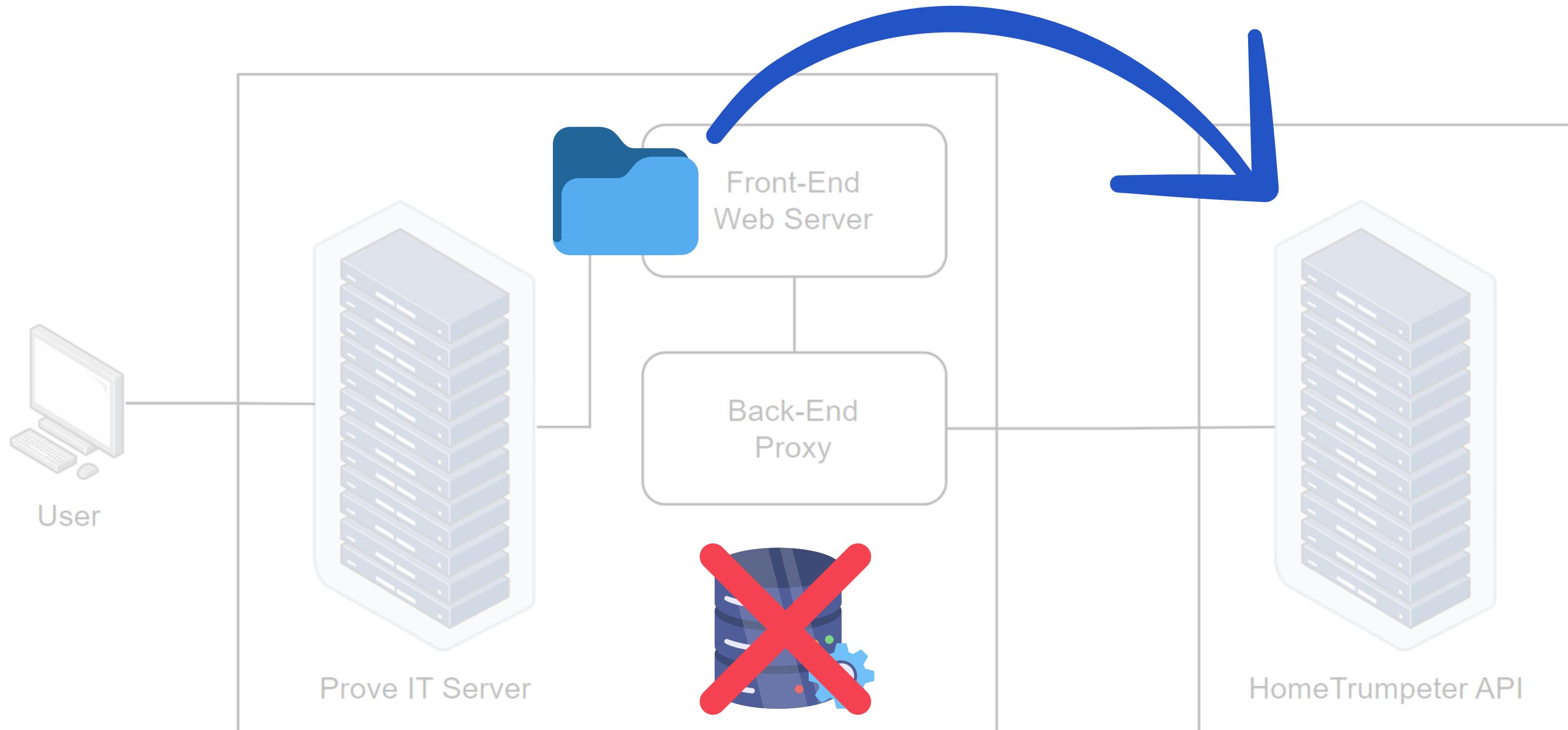
# Full-Stack



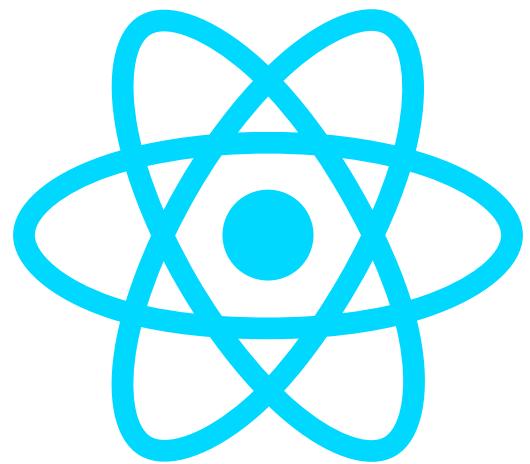
# Tech Stack



# Database



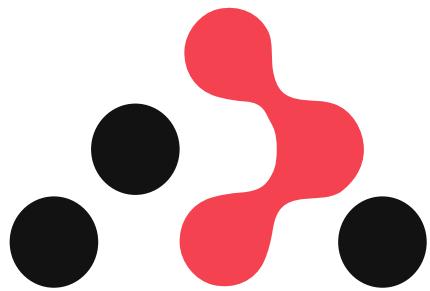
# Front-End



React JS



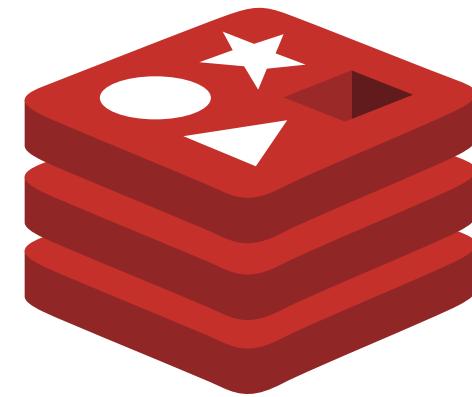
Bootstrap



**React Router**



# Back-End



Express  JS

---

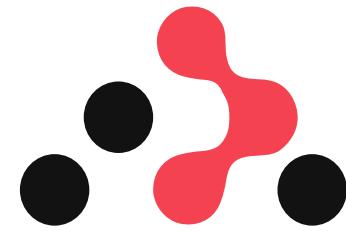
Express JS

Axios

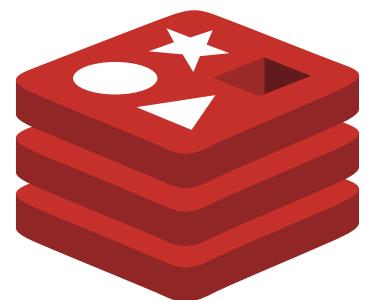
Node Mailer

Redis

# Dependencies



**React Router**



**AXIOS**

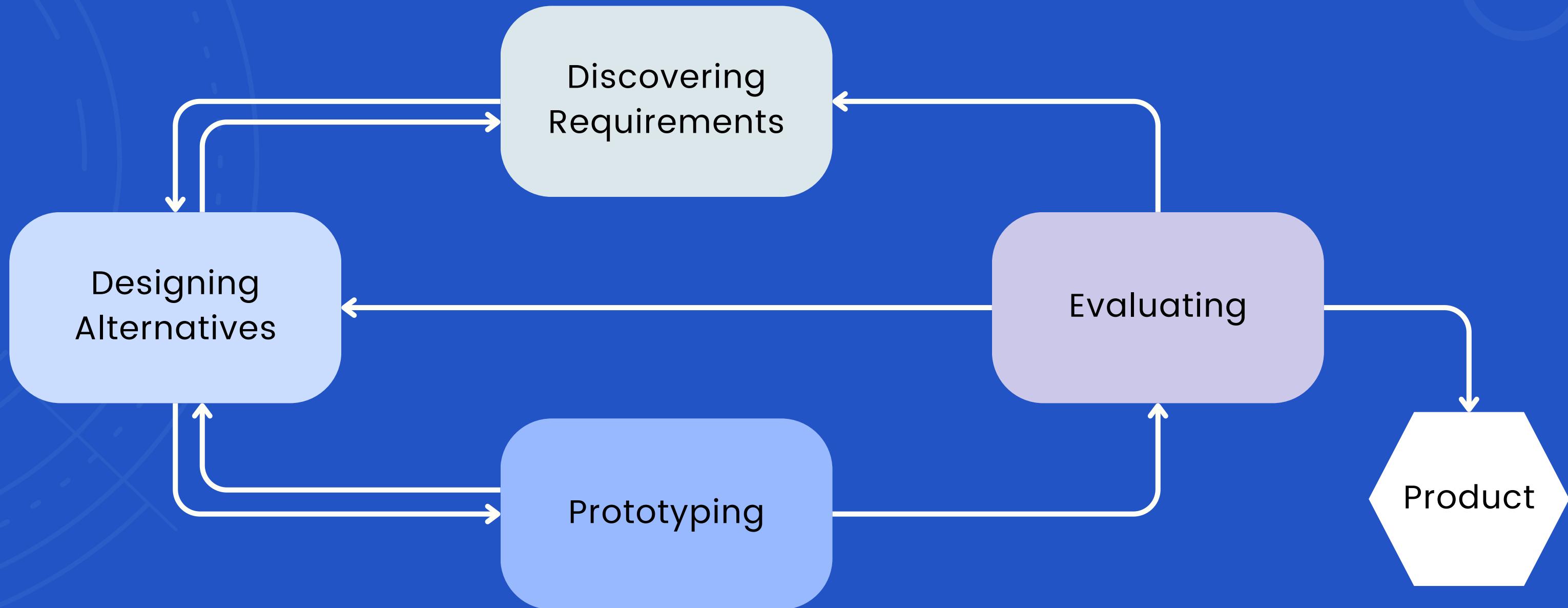


**HomeTrumpeter API**

**Frontend**

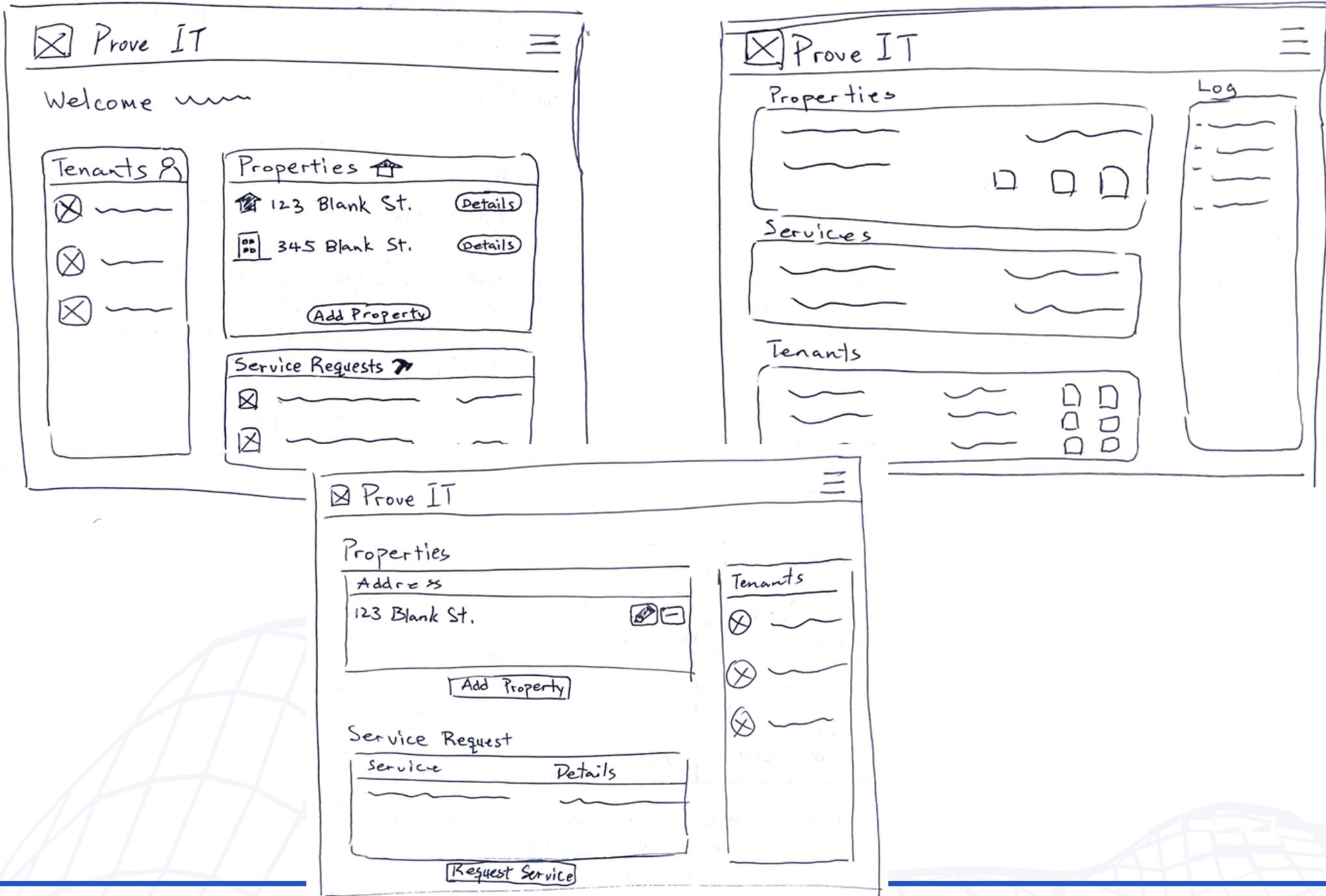
**Backend**

# User Interface



# Prototypes

Rapid Prototyping using User Stories



Entities	Task
Homeowner	As a homeowner, I would like to sign up for the services, so that I can add/save/edit my information.
	As a homeowner, I would like to add properties, so that I can keep track of them.
	As a homeowner, I would like to invite tenants, so that I can rent my properties to them.
	As a homeowner, I would like to approve tenants, so that I can ensure my tenants are all up to certain standards.
	As a homeowner, I would like to invite private service providers, so that I can assign them to fix issues with my properties.
Service Provider	As a homeowner, I would like to allocate tasks to service providers, so that I can track who is working on what.
	As a homeowner, I would like to add/save/edit the status of a ticket, so that I am as flexible as possible with my services.
	As a service provider, I would like to accept invitations from homeowners, so that I can provide service to them.
	As a service provider, I would like to sign up, so that I can add/save/edit my information.
	As a service provider, I would like to submit maintenance tickets, so that I can track my work.
Tenant	As a service provider, I would like to update the status of a ticket, so that I can notify the homeowner and tenant of the service progress.
	As a service provider, I would like to be able to mark my job as completed, so that I can finalize the work and get paid.
	As a tenant, I would like to accept invitations from homeowners, so that I can become a tenant at the respective property.
	As a tenant, I would like to sign up, so that I can add/save/edit my personal information.
	As a tenant, I would like to submit maintenance tickets, so that I can request services.
Homeowner	As a tenant, I would like to track the status of my tickets, so that I know the progress of my service request.
	As a tenant, I would like to confirm the completion of service requests, so that the invoice can be finalized.

# Prototypes

Hi-Fi Prototypes

Dashboard Homeowner

Properties

221 Hill Street, Milwaukee, WI

Add a property...

Service Requests

Service	Provider	Property
Sink and shower in...	Plumbing service +	221 Hill Stre...

Request a service

Tenants

Name	Property	Status
Bob Ross	221 Hill Street	Invite Sent

Add a tenant

HOME TRUMPETER

### Service Provider Sign up

First Name	<input type="text" value="Dave"/>
Last Name	<input type="text" value="0123456"/>
SIN	<input type="text" value="Bell MTS"/>
Company Name	<input type="text" value="Internet Service"/>
Service Type	<input type="text" value="18001234567"/>
Company Phone	<input type="text" value="bell@mts.ca"/>
Company E-mail	<input type="text" value="Canada"/>
Country	<input type="text" value="Winnipeg"/>
City	<input type="text" value="MB"/>

Request Sign-up

HOME TRUMPETER

### Service Request

Service Info	
Property	<input type="text" value="221 Hill Street, Milwaukee, WI"/>
Service Provider	<input type="text" value="Plumbing Service"/>
Service Information	<input type="text" value="Sink and shower in apartment 4 are not working and require repair."/>

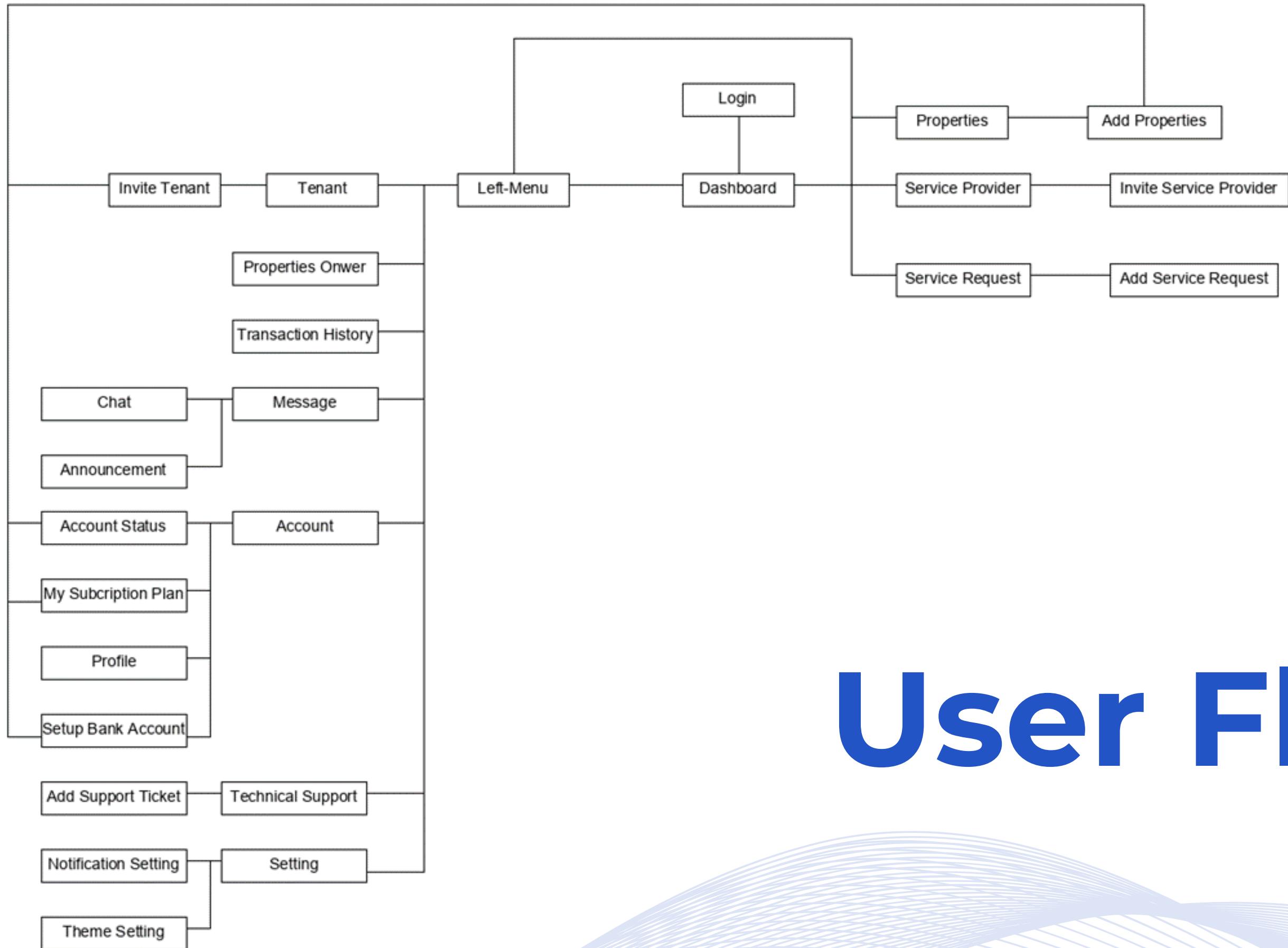
Submit Request

HOME TRUMPETER

### Invite Service Provider

Service Info	
Service Name	<input type="text" value="Bell MTS"/>
Service Type	<input type="text" value="Internet Service"/>
Phone Number	<input type="text" value="18001234567"/>
E-mail	<input type="text" value="bell@mts.ca"/>
Property	<input type="text" value="221 Hill Street, Milwaukee, WI"/>

Send Invite



# User Flow

# Design Principles



Prove IT by HomeTrumpeter works for you to make property management easier!



**Visibility** is established by maintaining a **brand kit**, which provides **consistent and recognizable** colour schemes, logos, icons, and fonts.

Ready to get started?

Log in

Sign up

The dashboard interface includes a login form with 'Remember me' and 'Forgot password?' links, a property management section with a table for adding properties, and a service requests section with a table for requesting services.

**Flexbox** and **grid** layouts in CSS are used to create **logical mappings**.

© 2024 HomeTrumpeter. All rights reserved.

Privacy Policy

Terms of Service

Contact Us

Remember me

[Forgot password?](#)



**Feedback** is provided by **hover effects** and **loading icons** when action is taken.

The property info form includes a house icon, a dropdown for selecting a state, and fields for property name, address, rent amount, and a submit button.

**External consistency** allows users to exploit **pre-existing knowledge** of similar systems, as well as hinting possible actions, which provides **affordances**.

# API

HomeTrumpeter has built a Restful API that includes features required to build a comprehensive property management system for property managers, homeowners, tenants, and service providers.

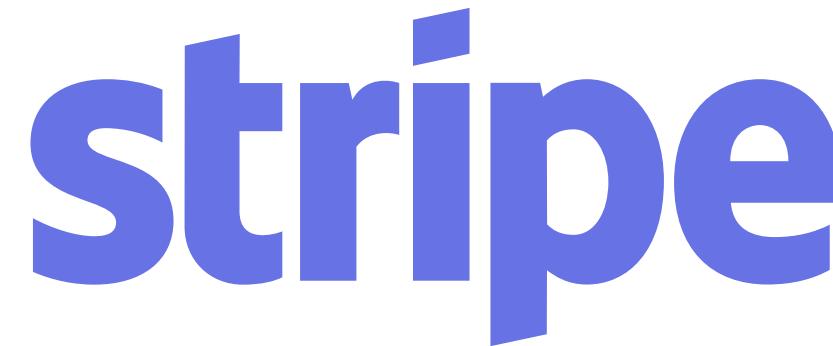
## HomeTrumpeter API

## Proxy



# HomeTrumpeter API

## Major Dependencies



## Categorization

- |                |                       |                      |
|----------------|-----------------------|----------------------|
| · user         | · settings            | · chat               |
| · verification | · template            | · video              |
| · customer     | · support-ticket      | · payment-processing |
| · files        | · property-management | · stripe             |
| · plan         | · service-provider    | · agreement          |
| · intro-slider | · ticket              | · background-check   |
| · location     | · job                 | · general            |
| · school       | · service-request     | · mail               |
| · admin        | · notification        |                      |

# JSON Web Token (JWT)

## Signature

JWT tokens used in the Prove IT system are digitally signed using a secret, which is stored ONLY in the HomeTrumpeter system.

The token is composed of:

- **Header**
- **Payload**
- **Signature**

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.

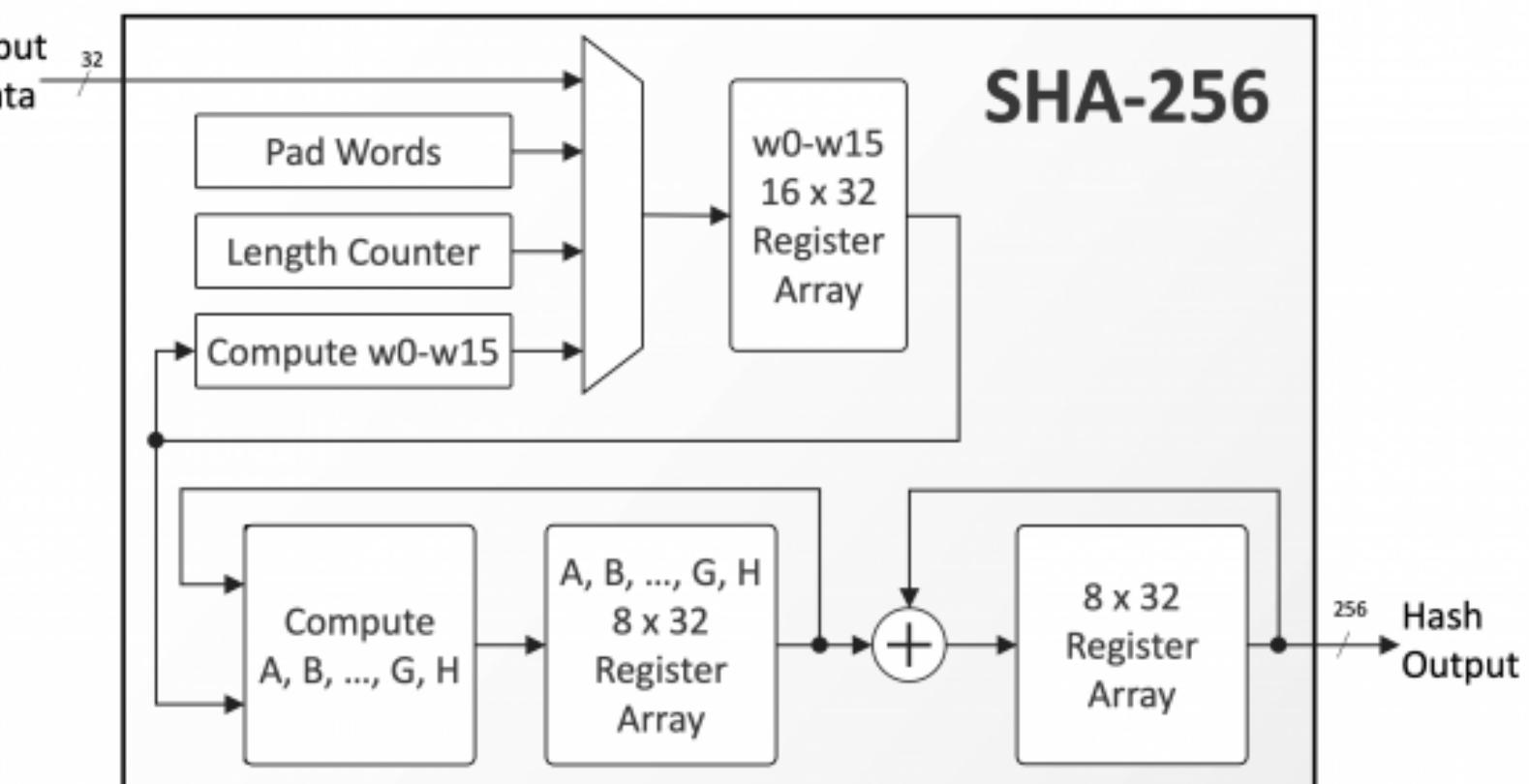
eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4

gRG9lIiwickXNTb2NpYWwiOnRydWV9.

4pcPyMD09o1PSyXnrXCjTwXyr4BsezdI1AVTmud2fU4

## SHA256

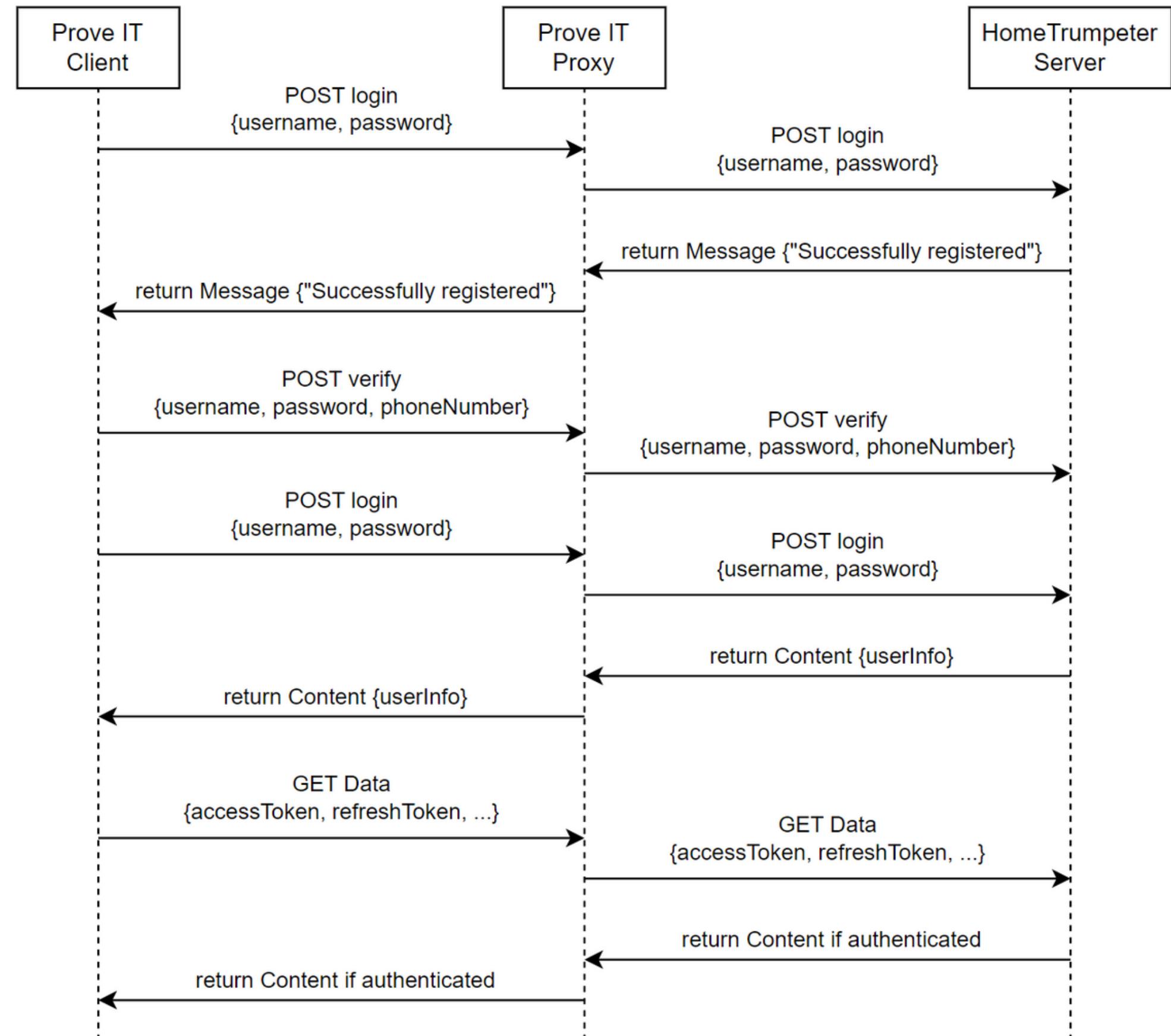
SHA256 encryption is used to ensure confidentiality of the signature, which is used to verify integrity of the information transferred.



Source: CAST, inc.

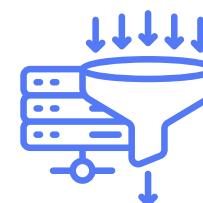
# JWT Session Management

Session management is implemented using the JWT tokens provided in the HomeTrumpeter API. The access tokens will be stored in localStorage of the web browser and retrieved for POST and GET requests.

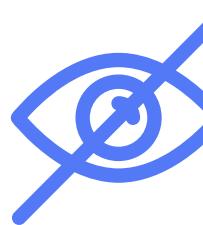


# Proxy

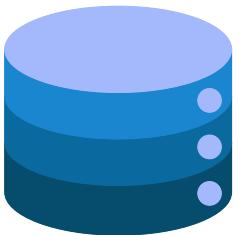
## Functions



Rate-Limiting



Hide Business Logic



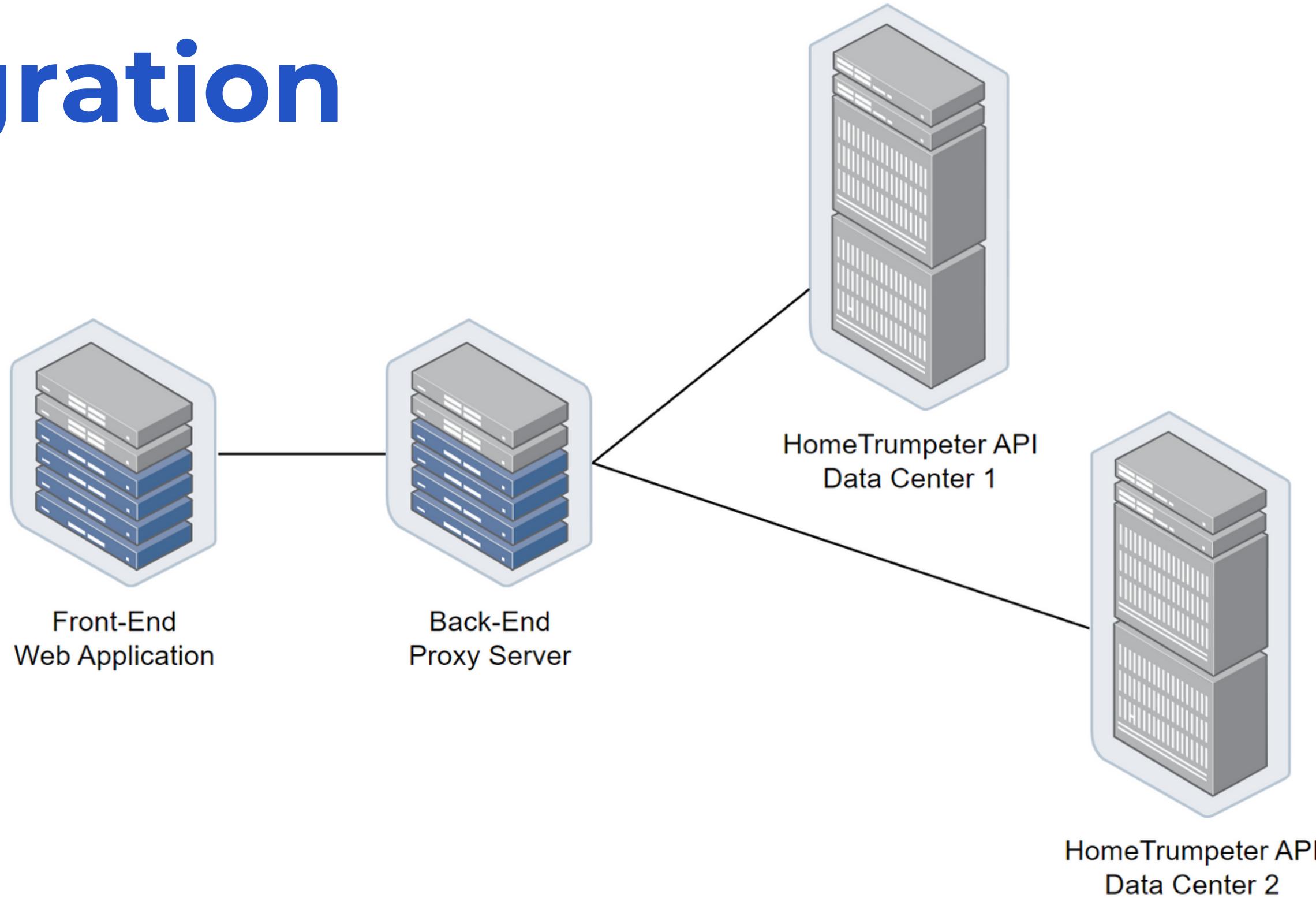
Caching

A common issue when using private API keys is including the API key in the source code or client-side HTTP requests. In order to avoid exposing the API key, a proxy server is used as a middle-man between the application and API server. This will redirect all API calls to the HomeTrumpeter API, along with the required authorization headers and bearer tokens.

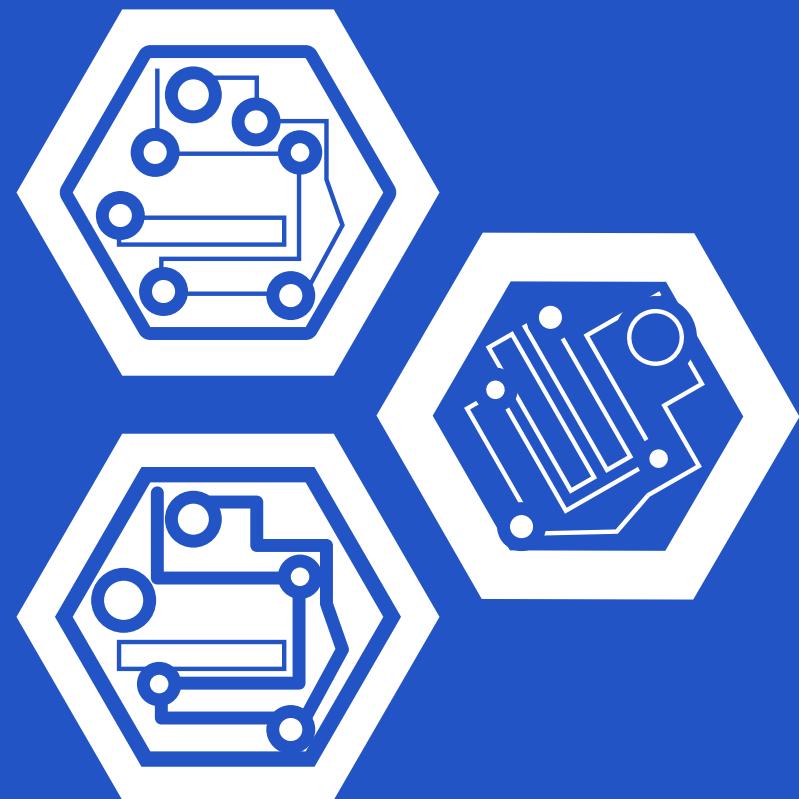
Furthermore, the proxy server will allow scaling of the system through methods such as rate-limiting, load-balancing, and traffic splitting. It also permits us to manipulate the API responses to include the minimal information needed by the client, thus exposing less surface for attackers to exploit.



# Integration



# Detailed System Design



The following slides will explain some specific reasons as to why certain components were chosen for this project

# Design Implications

## React JS

The front-end system used ReactJS as a foundational framework for building the user interfaces of the web application.

## React Router

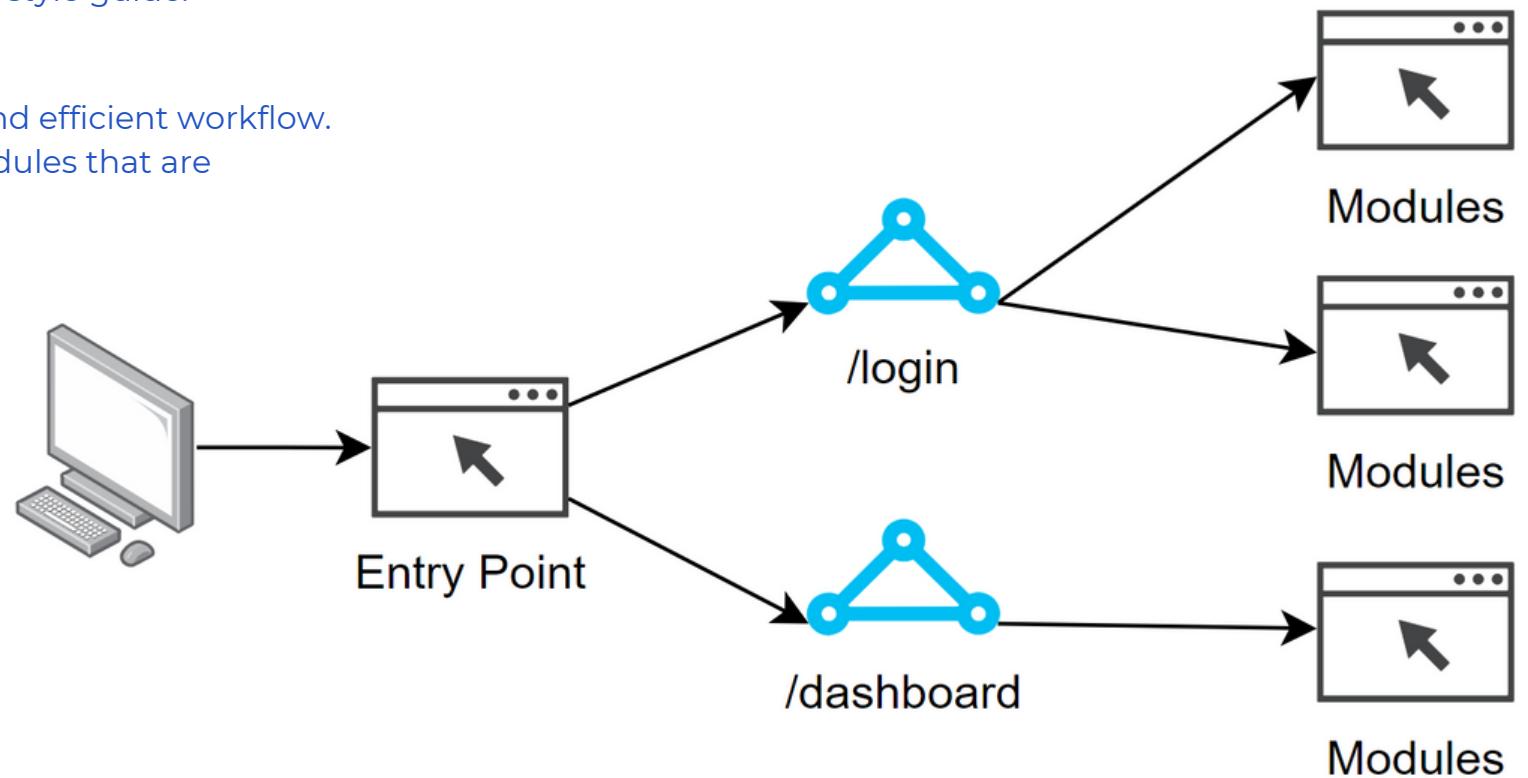
Routing was implemented using ReactRouter to point URLs to respective web pages. React contexts were used in conjunction to enforce user access to their respective permission levels.

## Bootstrap

Design of the UI was done using React Bootstrap, which includes functional components. These were further manipulated through custom CSS to fit the Prove IT style guide.

## Vite

Vite was used as a development server and bundler to ensure a quick and efficient workflow. This ensures that routes accessed by users are only required to load modules that are immediately used, which reduces loading latency significantly.



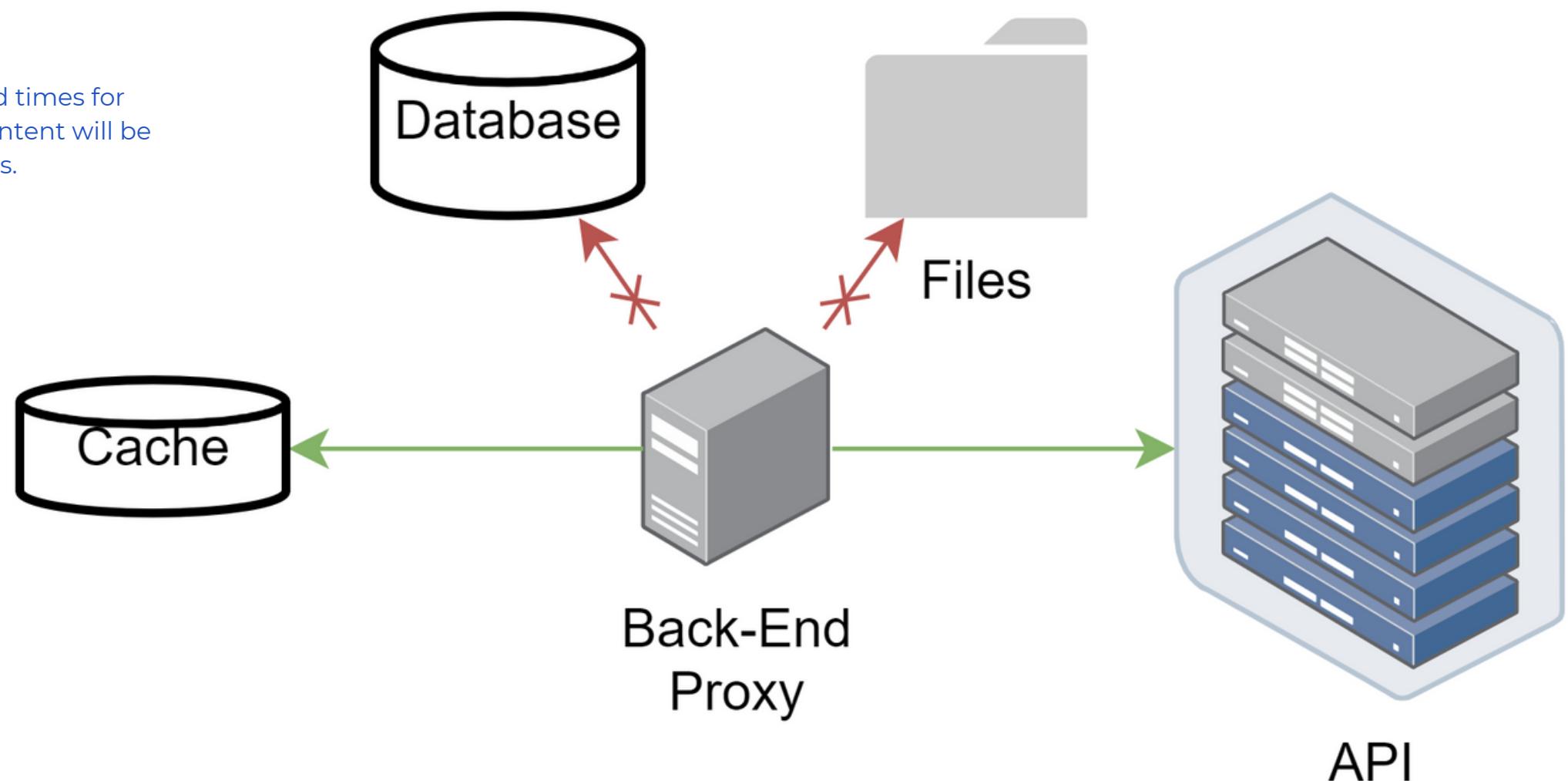
# Design Implications

## Express JS

The back-end system uses ExpressJS. Routing is implemented using the built-in router and components are functionalized to allow maximum reusability. No database or filesystem will be used, as the back-end will only serve as a proxy between the web application and the HomeTrumpeter API.

## Redis

Caching with Redis will significantly improve load times for frequently accessed pages and content. Static content will be served directly as to eliminate unnecessary delays.

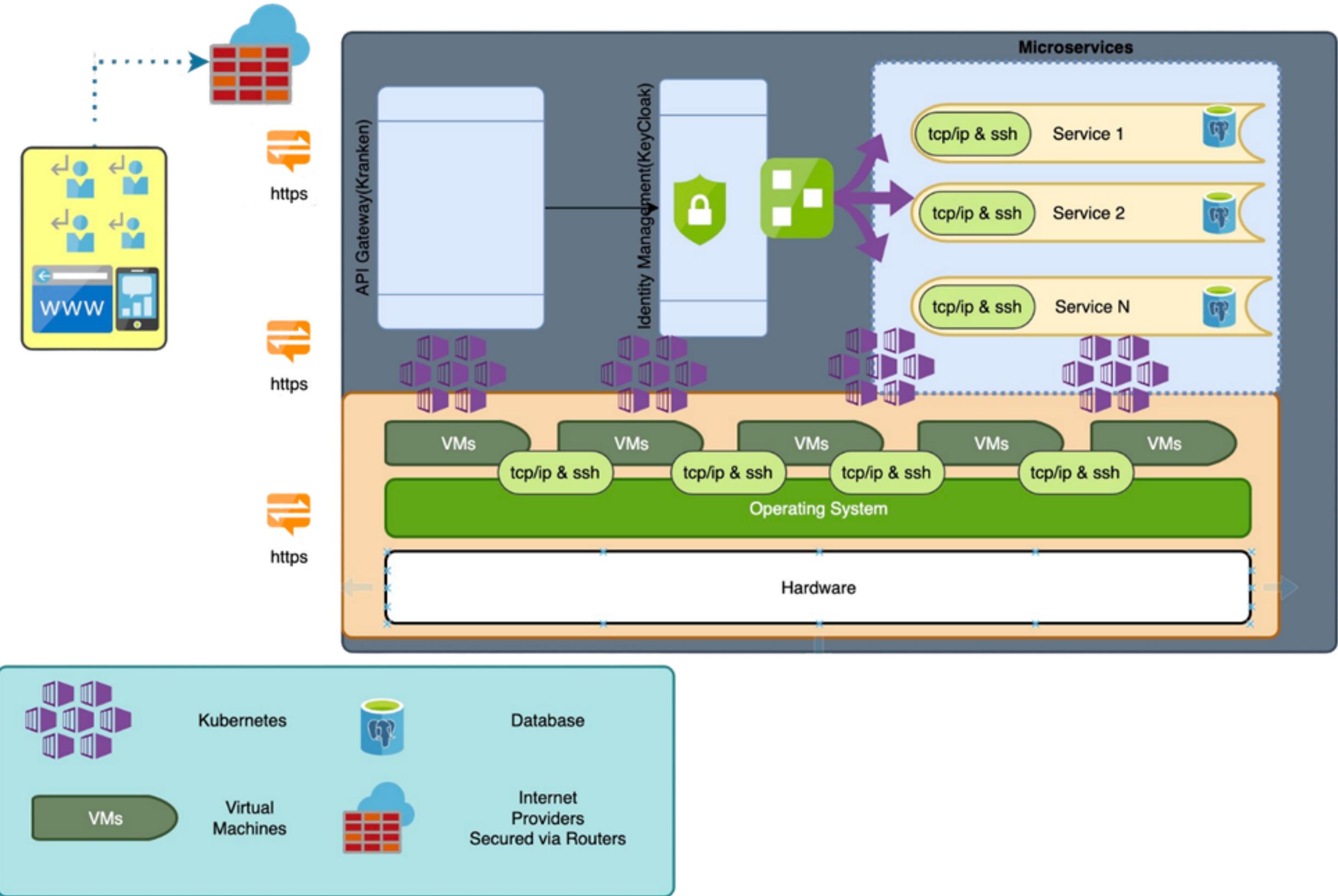


# Security

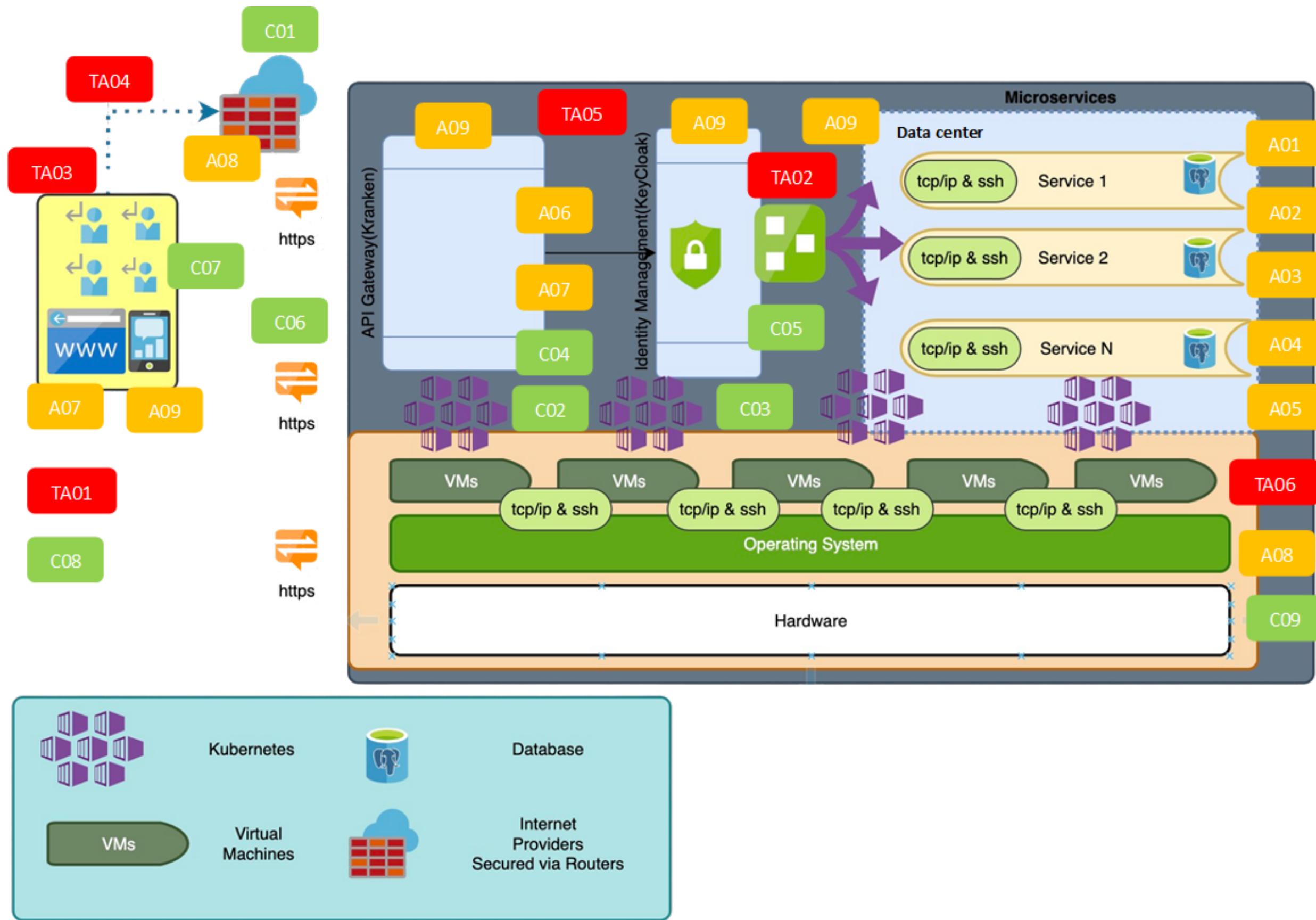


Security measures are implemented to ensure integrity of the system

# System

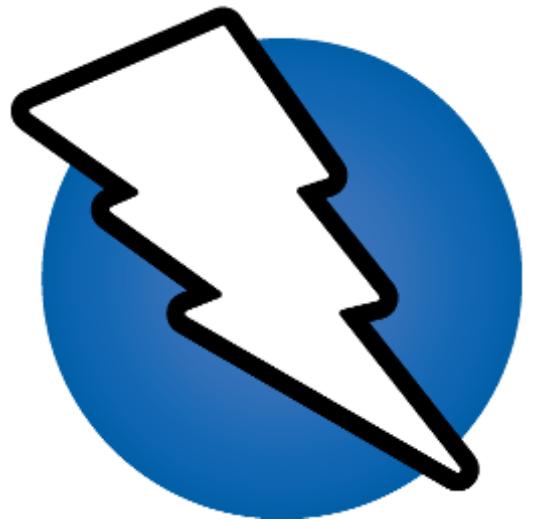


# Threats



# Detecting Vulnerabilities

OWASP ZAP



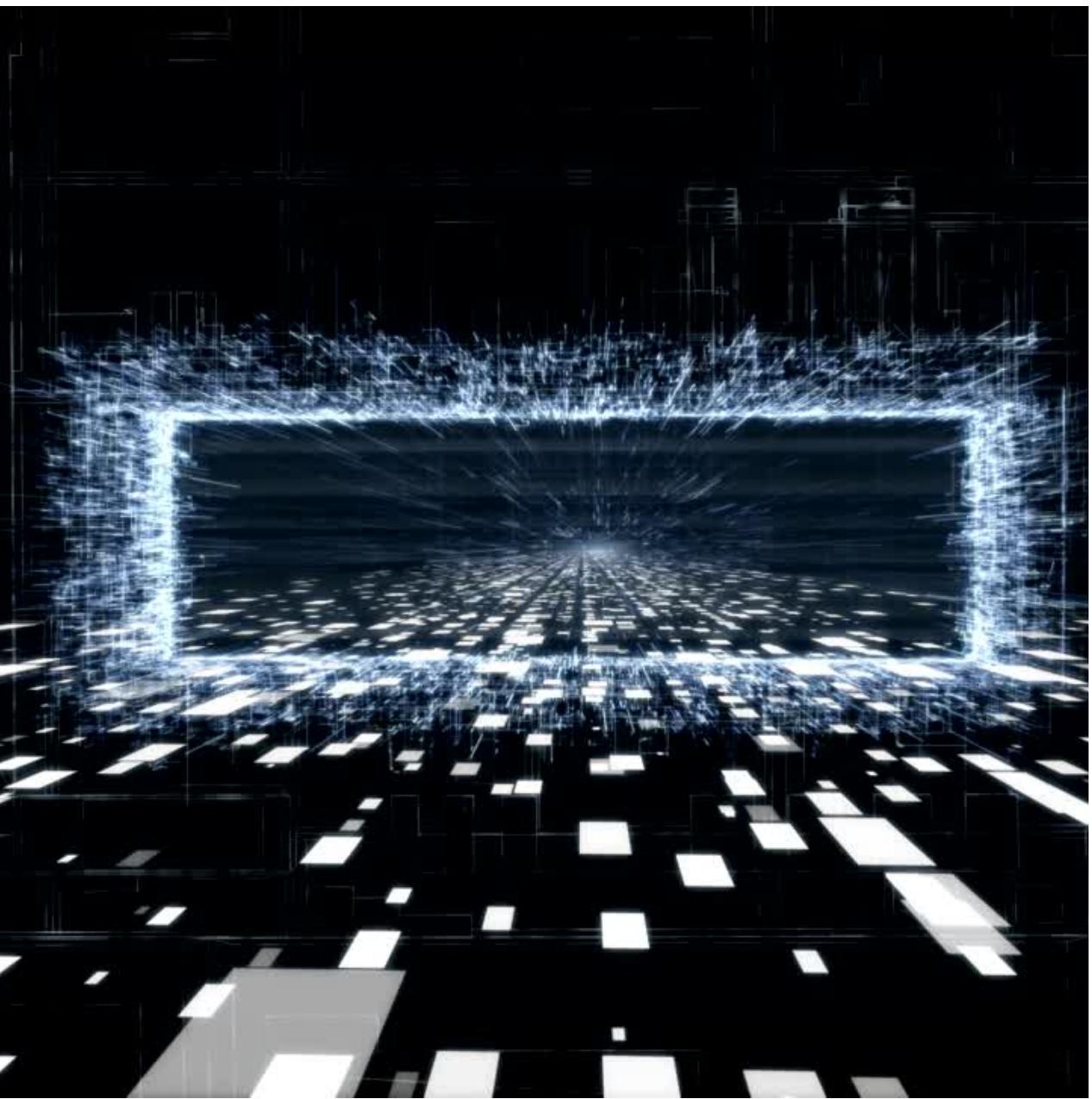
Open-source application penetration testing tool. Acts as a daemon between the user and application to intercept packages and analyze for vulnerabilities.

Snyk



# snyk

Vulnerability scanning for the entire deployment process. Covers source code, deployment scripts, infrastructure as code, etc.



# OWASP ZAP



# OWASP ZAP

The screenshot shows the ZAP interface with the following details:

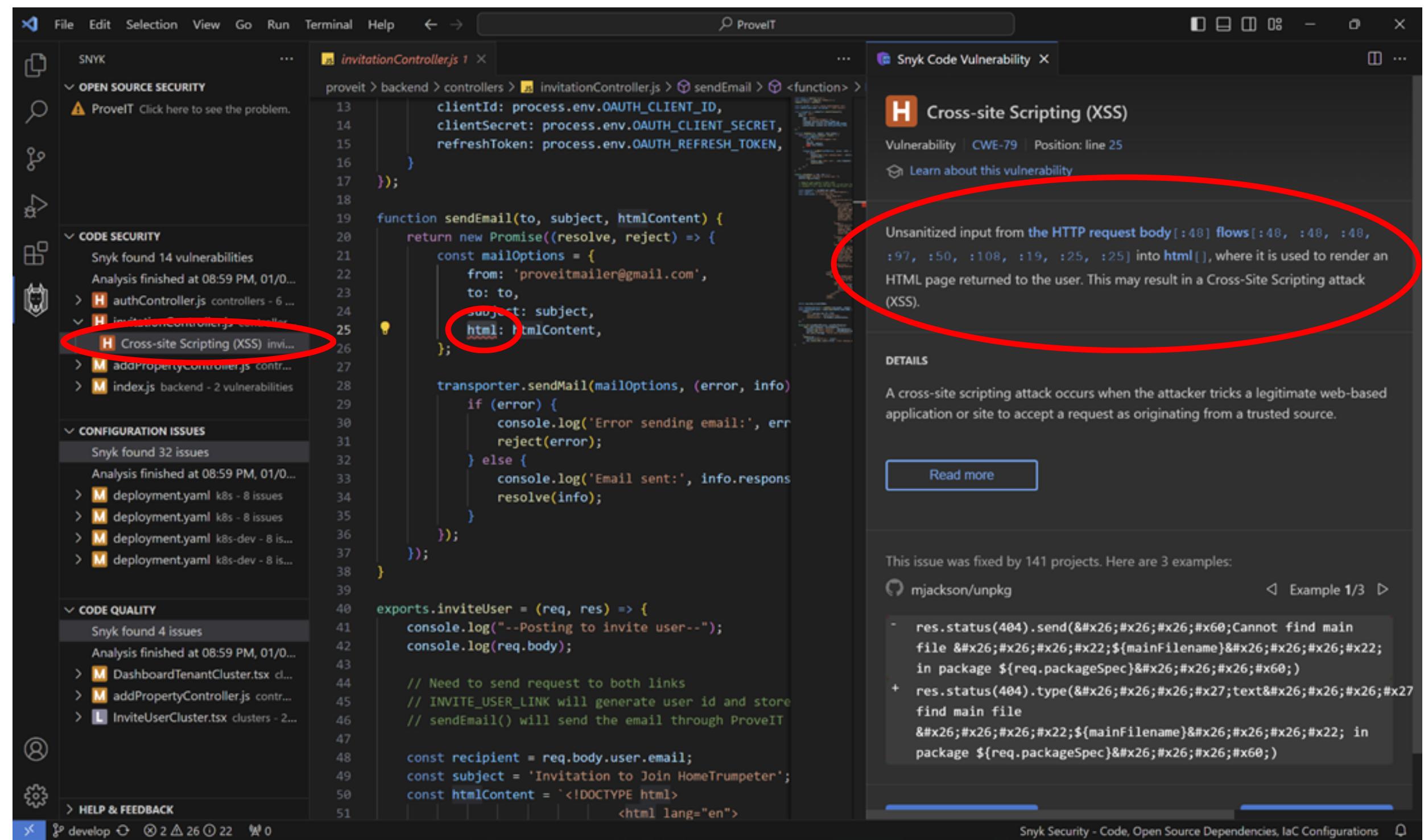
- Top Bar:** Untitled Session - ZAP 2.14.0, File, Edit, View, Analyse, Report, Tools, Import, Export, Online, Help.
- Left Sidebar:** ATTACK Mode, Sites (selected), Contexts (Default Context), and a list of sites including https://spocs.getpocket.com, http://localhost:5173, and https://private.canadianshield.cira.ca.
- Request/Response Tab:** Request tab is active, showing a successful HTTP/1.1 200 OK response with various headers like Access-Control-Allow-Origin, Content-Type, Cache-Control, Etag, Date, Connection, Keep-Alive, and Content-Length.
- Body Tab:** Response body shows a script tag that includes a module named "RefreshRuntime" and a script tag with a src attribute pointing to "/vite/client".
- Bottom Left:** History, Search, Alerts (selected), Output, WebSockets, Active Scan, AJAX Spider.
- Bottom Center:** Cloud Metadata Potentially Exposed alert details:
  - URL: http://localhost:5173/latest/meta-data/
  - Risk: High
  - Confidence: Low
  - Parameter:
  - Attack: 169.254.169.254
  - Evidence:
  - CWE ID: 0
  - WASC ID: 0
  - Source: Active (90034 - Cloud Metadata Potentially Exposed)
  - Input Vector:
  - Description: The Cloud Metadata Attack attempts to abuse a misconfigured NGINX server in order to access the instance metadata maintained by cloud service providers such as AWS, GCP and Azure. All of these providers provide metadata via an internal unrouteable IP address '169.254.169.254' - this can be exposed by incorrectly configured NGINX servers and accessed by using
- Bottom Right:** Cross Site Scripting (Persistent) - Prime, Cross Site Scripting (Persistent) - Spider, Cross Site Scripting (Persistent) - Oracle, SQL Injection, SQL Injection - MySQL, SQL Injection - Hypersonic SQL, SQL Injection - Oracle, SQL Injection - PostgreSQL, SQL Injection - SQLite, Cross Site Scripting (DOM Based), SQL Injection - MsSQL, Log4Shell, Spring4Shell, Server Side Code Injection, Remote OS Command Injection, XPath Injection, XML External Entity Attack, Generic Padding Oracle, Cloud Metadata Potentially Exposed, Server Side Template Injection, Server Side Template Injection (Blind), Directory Browsing, Buffer Overflow.

# Snyk

CWE-79

Cross-Site Scripting (XSS) attack vulnerability at **line 25** in the `invitationController.js` file.

Examples of solutions from other open-source projects on GitHub are shown.



The screenshot shows the Snyk IDE interface. On the left, the Snyk dashboard displays various security findings: 14 code security vulnerabilities (including the XSS issue), 32 configuration issues, and 4 code quality issues. The main workspace shows the `invitationController.js` file with the following code:

```
13     clientId: process.env.OAUTH_CLIENT_ID,
14     clientSecret: process.env.OAUTH_CLIENT_SECRET,
15     refreshToken: process.env.OAUTH_REFRESH_TOKEN,
16   });
17 }
18
19 function sendEmail(to, subject, htmlContent) {
20   return new Promise((resolve, reject) => {
21     const mailOptions = {
22       from: 'proveitmailer@gmail.com',
23       to: to,
24       subject: subject,
25       html: htmlContent,
26     };
27
28     transporter.sendMail(mailOptions, (error, info) =>
29       if (error) {
30         console.log('Error sending email:', error);
31         reject(error);
32       } else {
33         console.log('Email sent:', info.response);
34         resolve(info);
35       }
36     );
37   });
38 }
39
40 exports.inviteUser = (req, res) => {
41   console.log('--Posting to invite user--');
42   console.log(req.body);
43
44   // Need to send request to both links
45   // INVITE_USER_LINK will generate user id and store
46   // sendEmail() will send the email through ProveIT
47
48   const recipient = req.body.user.email;
49   const subject = 'Invitation to Join HomeTrumpeter';
50   const htmlContent = `<!DOCTYPE html>
51   <html lang="en">
```

The line `html: htmlContent,` is highlighted with a red oval. The right-hand panel, titled "Snyk Code Vulnerability" and "Cross-site Scripting (XSS)", provides a detailed description of the vulnerability: "Unsanitized input from the HTTP request body[:48] flows[:48, :48, :48, :97, :50, :108, :19, :25, :25] into html[], where it is used to render an HTML page returned to the user. This may result in a Cross-Site Scripting attack (XSS)." A red oval also encircles this panel. Below it, a "Read more" button and a section showing examples from GitHub are visible.

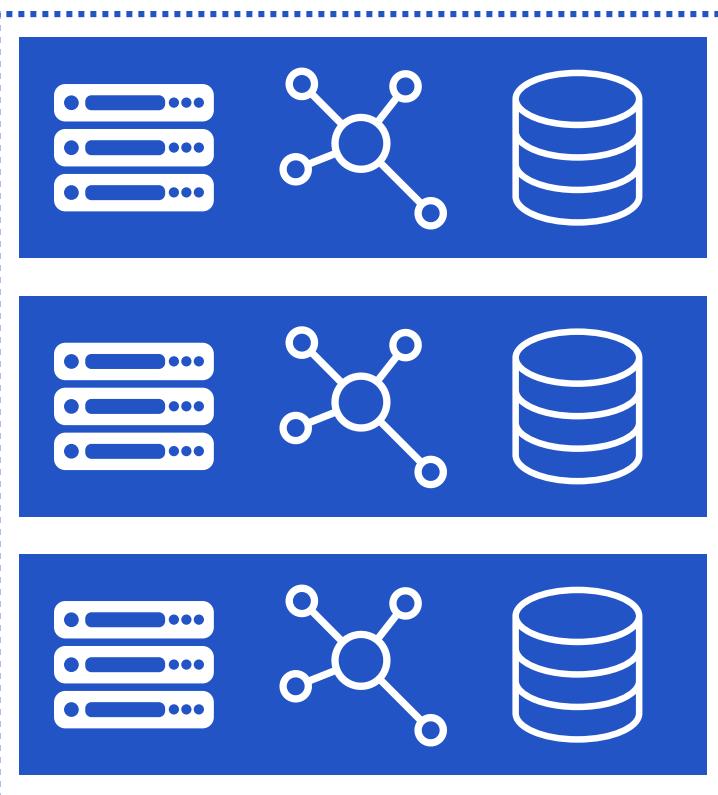
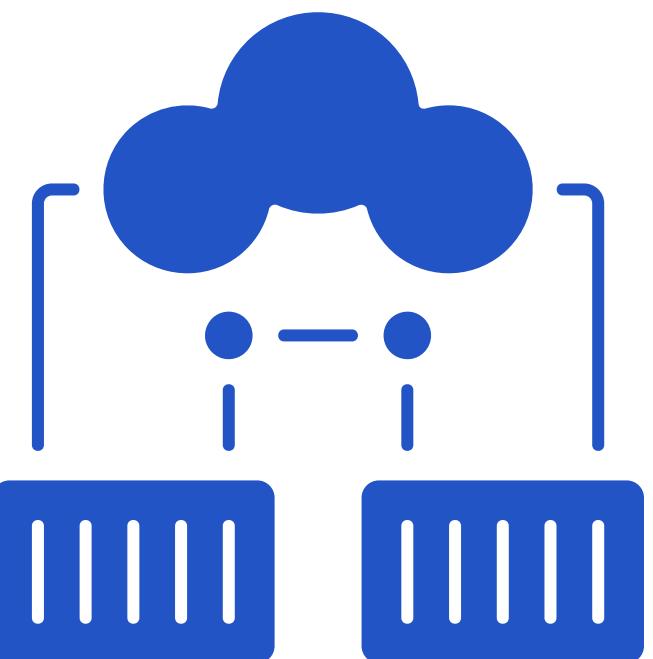
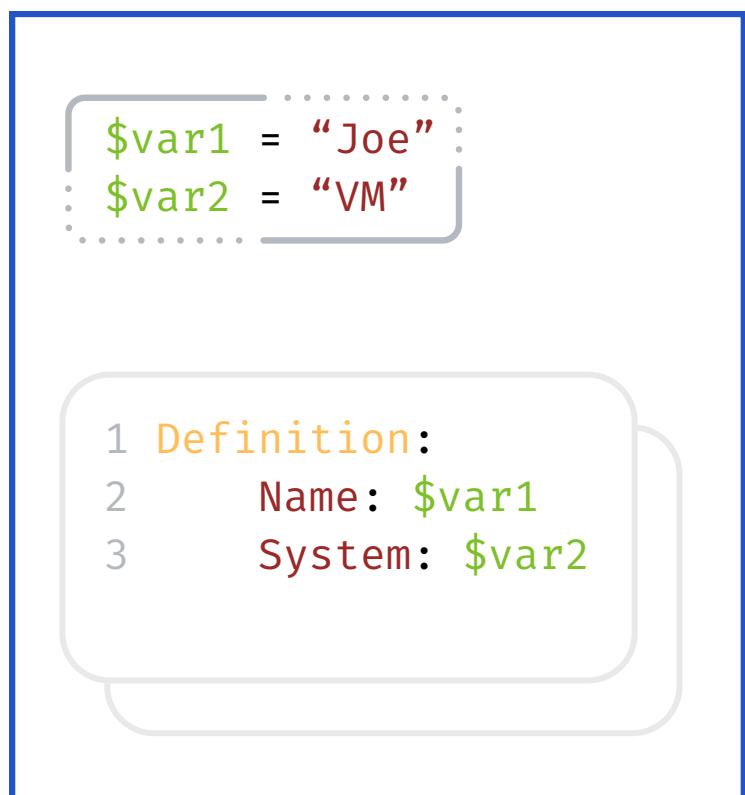
# Testing



Various testing strategies are implemented to ensure a robust system is delivered to the user, without risk of failure.

# Deployment

## Infrastructure as Code (IAC)

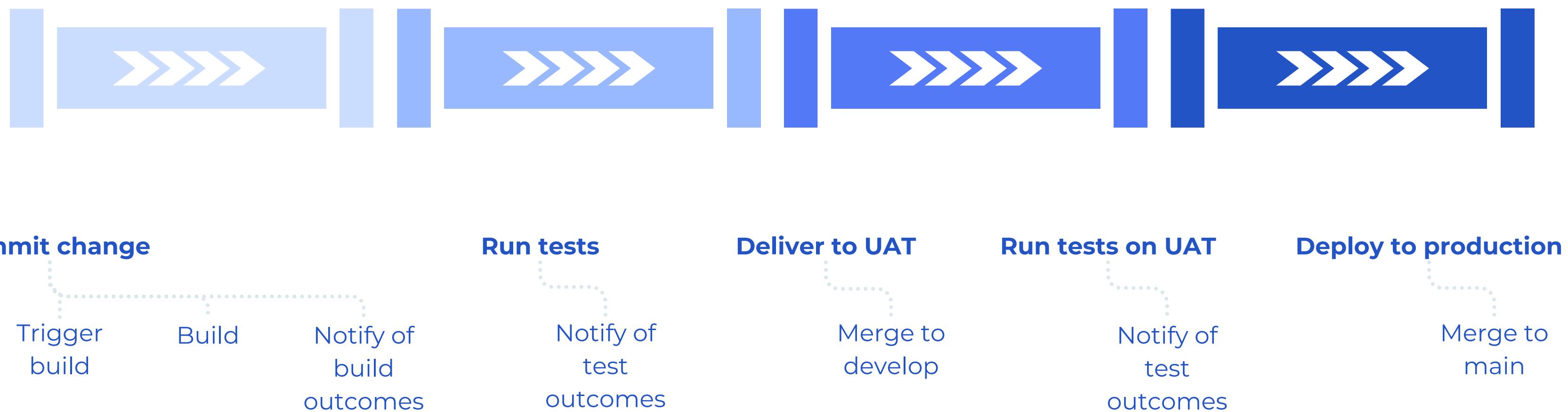


**Adjust parameters** for each deployment

Package and ship to the cloud

Deploy in **isolation** and test in simulated real-world scenarios

# Pipeline



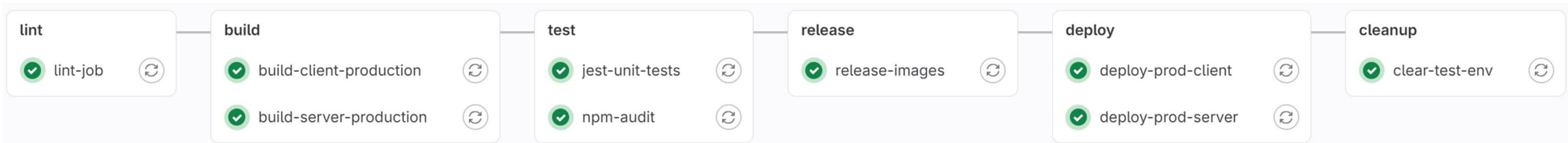
# Pipeline

## Merge branch 'develop' into 'main'

✓ Passed Arshjot Ghuman created pipeline for commit `ec0d9012`  finished 3 weeks ago

For `main`

latest  9 Jobs  9.9  7 minutes 16 seconds, queued for 1 seconds



# Testing Strategies



## Unit Testing

Validate individual components, functions, or units of code

Test-Driven Development (TDD) using Jest

QA team writes tests for functional components

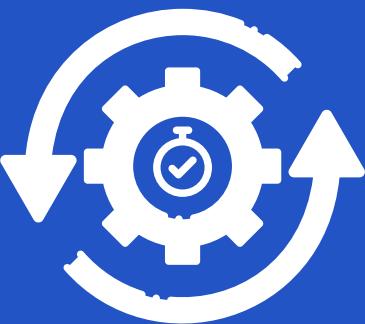


## Integration Testing

Verifies the interactions and compatibility between different components

Simulating real-world usage scenarios

Ensures that various parts of the system work harmoniously together



## Regression Testing

Re-running previously executed tests

Detect and prevent the introduction of new bugs or issues as code evolves

# Testing Strategies



## User Acceptance Testing

Last stage of testing where change requests are accepted

Users perform real-world tasks in real-world environments and provide feedback



## End-to-End Testing

Simulate user interactions with the application

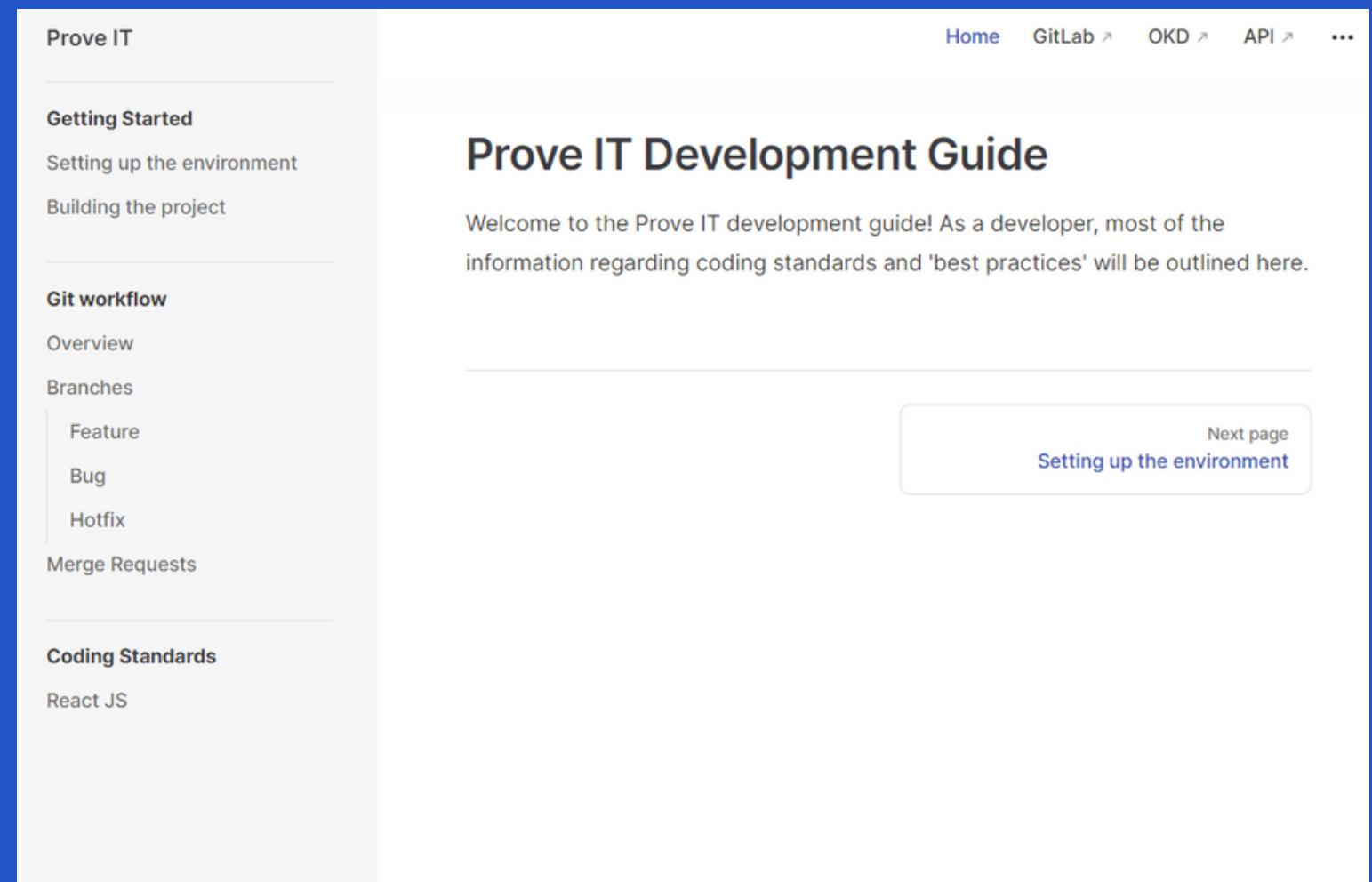
Check the entire flow from start to finish

# Documentation

Comprehensive documentation is provided to the development team to ensure established standards are met. Coding documentation ensures consistent codebase for ease of maintenance. Deployment documentation ensures robustness of the delivered product.

## Coding Standards

## Deployment Instructions



The screenshot shows a website layout for 'Prove IT Development Guide'. The header includes a logo, navigation links for 'Home', 'GitLab', 'OKD', 'API', and '...', and a search bar. The main content area has a title 'Prove IT Development Guide' and a welcome message. To the left is a sidebar with a navigation menu:

- Prove IT**
- Getting Started**
  - Setting up the environment
  - Building the project
- Git workflow**
  - Overview
  - Branches
    - Feature
    - Bug
    - Hotfix
  - Merge Requests
- Coding Standards**
  - React JS

At the bottom right of the main content area is a button labeled 'Next page' with the text 'Setting up the environment'.

# Coding

Prove IT developers have access to documentation that outlines coding and integration standards. Coding standards enforce basic syntactic patterns to guarantee a uniform code base. Functional programming guidelines are also in place for ReactJS development to increase reusability of various components.

The sidebar contains the following navigation links:

- Getting Started**
  - Setting up the environment
  - Building the project
- Git workflow**
  - Overview
  - Branches
    - Feature
    - Bug
    - Hotfix
  - Merge Requests
- Coding Standards**
  - React JS

## Coding Guidelines - ReactJS

### Table of Contents

- [1. Basic Rules](#)
- [2. Naming](#)
- [3. Declaration](#)
- [4. Alignment](#)
- [5. Quotes](#)
- [6. Spacing](#)
- [7. Props](#)
- [8. Parentheses](#)
- [9. Tags](#)
- [10. Methods](#)
- [11. Ordering](#)

### Basic Rules

- Only include one React component per file.
- Always use JSX syntax.
- Do not use `React.createElement` unless you're initializing the app from a file that is not JSX.

### Naming

- File- and component name need to be identical.
- Use PascalCase naming convention for filename as well as component name, e.g. `GlobalHeader.js`

```
// Bad  
// Filename: foo.js
```

javascript

### On this page

- Table of Contents
- Basic Rules
- Naming
- Ordering
- Alignment
- Quotes
- Tags
- Stateless function components
- PropTypes declarations
- Prefixing none React methods
- Prefixing component wide varia...
- Using handler methods
- Using "container" component...
- Closing Components without ch...
- List iterations
- Formatting Attributes
- Inline CSS styles
- Use "classnames" to set CSS cl...
- Working with DOM listeners
- Using StaticContainer for more ...
- Use higherOrder functions to a...
- Sources

# Deployment

A workflow for Git integration is provided to embrace the idea of “infrastructure as code”, enabling the team to easily test functionality and integration prior to delivering the source code.

Prove IT

---

**Getting Started**

- Setting up the environment
- Building the project

---

**Git workflow**

- [Overview](#)
- [Branches](#)
- [Feature](#)
- [Bug](#)
- [Hotfix](#)
- [Merge Requests](#)

---

**Coding Standards**

- React JS

## Git Workflow

### Branches

#### Permanent Branches

Purpose	Branch Prefix	Description
Production	main	Accepts merges from 'develop' branch
Development	develop	Accepts merges from 'feature', 'bug', and 'hotfix' branches

#### Temporary Branches

Purpose	Branch Prefix	Example
Adding features	feature-<desc>	feature-phone-verification
Fixing bugs/issues	bug-<desc>	bug-login-auth
Hotfix	hotfix-<desc>	hotfix-

### Primary Branches

The main repository will always hold two evergreen ready branches:

- [main](#)
- [develop](#)

The main branch `origin/main` should always represent the latest code deployed to production. During day to day development, the stable branch will not be interacted with.

As a developer, you will be branching and merging from `develop`. The branch `origin/develop` should be where the source code of `HEAD` always reflects a state with the latest delivered development changes for the next release.

When the source code in the `develop` branch is stable and has been deployed, all the

Home GitLab OKD API

---

On this page

- Branches
- Primary Branches
- Supporting Branches
- Workflow Diagram

# Thank You

We are here to “Prove IT” is possible to develop an API economy to revolutionize property management with a cutting-edge and easy-to-use system for property managers, homeowners, tenants, and service providers.



**The Ultimate Property Management System**