



THE UNIVERSITY OF  
**WINNIPEG**

**Dept. of Applied Computer Science**

**ACS-4901**

**Senior Systems Development Project**

**Prove IT – System Study Review**

**Team Members**

<u>Name</u>	<u>ID</u>
Arshjot Ghuman	3114007
Liam Kristjanson	3116156
Usmaan Sahar	3065852
Vi Le	3119710
W A Shadman	3132520
Xiao Zhang	3131956

**IS Director:** Victor Balogun

**Date:** 13 November 2023

# Table of Contents

- 1. Executive Summary..... 3
- 2. Project Overview..... 3
- 3. Agile Methodology Selection ..... 4
- 4. Project Scope ..... 5
  - 4.1 Medium Fidelity Figma Prototype ..... 5
  - 4.2 GitLab CI/CD Pipeline..... 5
  - 4.3 React Web Application..... 5
  - 4.4 (Time Permitting) React Native Mobile App ..... 7
  - 4.5 (Time Permitting) Machine Learning Model to Predict Service Provider Behavior ..... 7
- 5. Project Team and Roles..... 8
- 6. Timeline and Sprints ..... 10
- 7. Product Backlog..... 13
- 8. Sprint Planning and Execution..... 13
- 9. Daily Standups..... 14
- 10. Continuous Integration and Testing..... 14
- 11. Retrospectives and Continuous Improvement ..... 16
- 12. Risk Management..... 17
- 13. Budget and Resources ..... 18
  - 13.1 Allocation of Budget and Resources..... 18
  - 13.2 Tool Selection..... 19
- 14. Benefits and Expected Outcomes..... 20
- 15. Design Principles and Expectations..... 21
- 16. Appendix..... 22
  - 16.1 Product Backlog and Work Breakdown Structure ..... 22
  - 16.2 Burndown Charts..... 23
  - 16.3 User Interface Mock-ups (From Figma Prototype)..... 24
  - 16.4 Abuse Cases and Countermeasures ..... 27
  - 16.5 Architectural Diagram and Threat Model ..... 28
  - 16.6 Penetration Testing Plan and Implementation ..... 32

# 1. Executive Summary

HomeTrumpeter LLC is a company based in Illinois, USA. HomeTrumpeter has innovated a novel approach to property management through a centralized platform connecting tenants, homeowners, and service providers. To expand the reach of their product, they set out to enter the API economy. This would allow other developers to leverage the existing business logic of the HomeTrumpeter system, in exchange for a service fee. Our team is tasked to construct a new platform built on the foundations of the HomeTrumpeter API gateway, to "Prove IT" can be done.

Our system will allow homeowners to add their properties and invite tenants to the platform. Invited tenants will then be able to submit requests for required maintenance on the property. Upon receipt of a service request, a homeowner will be able to browse local service providers and seek a quote for the maintenance work.

Through this project, HomeTrumpeter will be able to stress-test each component of their microservices in a real-world scenario and measure the viability of constructing an independent system leveraging their API.

# 2. Project Overview

The goal of this project is to develop a platform that wraps around the HomeTrumpeter API gateway.

We will design and build a web interface in React, integrating into the HomeTrumpeter API gateway. A CI/CD workflow will be developed using GitLab to streamline the testing and deployment processes, allowing for fast and safe releases. The app will be continuously deployed to publicly accessible clusters managed by HomeTrumpeter through the OpenShift Kubernetes Platform.

Our system may not be used to replace the existing HomeTrumpeter web app. The primary objective of the project is to evaluate the feasibility of the API created by HomeTrumpeter. By working closely with the team at HomeTrumpeter, our team will be able to identify points of strength, or areas of improvement in the existing API. From this point, HomeTrumpeter will be able to better tailor their API to the requirements of an independent platform.

The API economy has become a massively important sector of the tech industry in recent years. Companies such as Amazon, Meta, Google, and Microsoft all take advantage of this economy by offering their services through an API gateway. This economy can offer a newfound stream of business for HomeTrumpeter, who have already developed the required services for a property management

system through the creation of their flagship product. By allowing other systems to leverage their services, HomeTrumpeter will be able to generate a new stream of income from a system they have already developed.

Our primary focus will be the construction of a web app leveraging the existing API. If time permits, we will explore the development of new features not included in the current HomeTrumpeter platform. Such features could include a React Native mobile app available on iOS and Android, or a machine learning model to predict service provider behavior using publicly available information.

### 3. Agile Methodology Selection

As an Agile group, our team will be using the Scrum framework to approach this project. Scrum is a framework allowing teams to self-organize towards a common goal, focusing on continuous experimentation, learning, and process improvement. The Scrum framework calls for work to be divided among fixed-length increments of time called sprints. Each sprint is furnished with a sprint planning meeting, several standup meetings, a sprint demo, and a retrospective meeting.

Our team decided on a sprint duration of two weeks to allow sufficient time for significant progress to be made each sprint, while still incorporating regular feedback from sponsors. Every sprint starts with a sprint planning meeting. In this meeting, the development team and stakeholders come together to agree on a set of goals to be completed in the upcoming sprint. Such a meeting allows for specific requirements to be clarified, and work items to be prioritized according to the needs of the project and the priorities of the stakeholders. The selected work items for the sprint form the “Sprint Backlog” which encompasses all work required to reach the goals set for the sprint.

Once a sprint backlog has been defined, the team meets regularly throughout the sprint for standup meetings. Standup meetings allow each member to share with the other members of the team the progress they have made, their plan for the upcoming days, and ask for help as needed.

At the conclusion of the sprint, the team meets with stakeholders for the sprint demo. In this meeting, the team demonstrates the work they have completed during the previous sprint and solicits feedback on the current state of the product. Following the demo, the team meets amongst themselves for a retrospective meeting. The sprint retrospective allows team members to candidly share any notable success they had, any challenges they faced, or lessons they learned during the previous sprint. This sharing of knowledge will allow the team to collectively learn from each member’s experiences, and improve our processes based on lessons learned.

Approaching a software project as a newly formed team using unfamiliar technologies brings with it a high level of uncertainty. The Scrum framework is specifically designed to handle uncertainty well. The continuous cycle of feedback offered by the sprint demo meetings allows us to ensure that the product we are developing meets the needs of the stakeholders. Committing to a set of goals every two weeks and seeing them through to completion allows us to manage uncertainty relating to our development output. Rather than performing extensive planning and documentation ahead of development, the Scrum framework allows us to develop in small increments starting very early in the project. This approach allows us to gain experience with the technologies we are using earlier, and better understand what will be possible to deliver with the given timeframe and resources.

## 4. Project Scope

Project "Prove IT" aims to develop a platform interfacing with HomeTrumpeter's API gateway. The scope encompasses the following key deliverables:

### 4.1 Medium Fidelity Figma Prototype

The team is developing a medium fidelity prototype to conceptualize and communicate our vision for the system. Such a prototype will allow our team to demonstrate the core functionalities of the system and solicit feedback from sponsors. The medium level of fidelity will allow us the flexibility to easily change UI designs based on stakeholder feedback. The team will then be able to use this prototype as a guide for interface design throughout the development process. As functionality is added to the system, the team will iterate on the prototype to conceptualize and communicate new features.

### 4.2 GitLab CI/CD Pipeline

To facilitate continuous deployment of changes to the system and ensure product quality, the team has begun to develop a continuous integration / continuous deployment pipeline in GitLab. The continuous integration portion of the pipeline will allow our team to automate unit tests, functional testing, static code analysis, dependency scanning, and fuzz testing. These functionalities will improve the quality and security of the delivered product, by automatically identifying potential issues with every push of new code to the repository. The continuous deployment portion of the pipeline will automate the process of building our application, containerizing it through Docker, and deploying it to HomeTrumpeter's private cloud infrastructure.

### 4.3 React Web Application

Commented [XZ1]: Can we change the number of these subheading to 4.1, 4.2, ..., to show it's under the 4th point

Commented [AG2R1]: Yes!

The primary deliverable of the project will be a React web application leveraging HomeTrumpeter's API. The planned user groups and their respective user stories are as follows:

#### I. Homeowners / Property Managers

- i. Create Account: Enable homeowners and property managers to create an account on the system. As a homeowner, I want to create an account on the HomeTrumpeter system so that I can manage my properties and tenants in a centralized manner.
- ii. Add Properties: Add the properties they own to the system. As a homeowner, I want to add my properties to the system, so that I can invite my tenants and create service requests for the property.
- iii. Invite Tenants: Invite tenants renting their properties to the system. As a homeowner, I want to invite my tenants to the HomeTrumpeter system so that they can communicate with me and create service requests for the property.
- iv. Invite Service Provider: Invite known service providers to the system so that they may send service requests to them. As a homeowner, I want to invite a known service provider to the HomeTrumpeter system so that I may dispatch them for service requests.
- v. Request quote: Request a quote from a service provider for a required maintenance job on their property. As a homeowner, I want to request a quote from a service provider so that I know their estimated rate and when they are available for a service request.
- vi. Approve quote: Approve a received quote from a service provider for a maintenance job. As a homeowner, I want to approve a received quote from a service provider so that they can provide the required maintenance work for the agreed-upon rate.

#### II. Tenants

- i. Accept Invitation: Accept an invitation to the platform sent by their landlord. As a tenant, I want to accept an invitation from my landlord so that I can create an account on the HomeTrumpeter system.
- ii. Sign up: After accepting an invitation to the system, a tenant should be able to create their own account. As a tenant, I want to sign up for an account on the HomeTrumpeter system so that I can connect with my landlord and make requests for required maintenance work.
- iii. Submit Maintenance Request: Submit a request for required maintenance, notifying the homeowner to seek a service provider to complete the job. As a tenant, I want to submit a maintenance request for required work so that my landlord can be aware of the needed maintenance and dispatch an appropriate service provider.

- iv. Track Request Status: Track the status of a service request, and view when it is expected to be completed. As a tenant, I want to track the status of a service request I have created so that I can prepare for the arrival of the service provider and know when the service will be completed.

### III. Service Providers

- i. Accept Invitation: Accept an invitation to the platform sent by a homeowner or property manager. As a service provider, I want to accept an invitation from a Homeowner to the HomeTrumpeter system so that I can create an account to access the system.
- ii. Sign up: After accepting an invitation to the system, a service provider should be able to create their own account. As a service provider, I want to create an account on the HomeTrumpeter system so that I can be dispatched for service requests by homeowners on the system.
- iii. Provide quote: In response to a service request from a homeowner or property manager, a service provider will be able to provide a quote for the maintenance work and offer dates they are available for the job. As a service provider I want to send a quote to a homeowner in response to a request from a homeowner so that we can agree on a date and rate for the service to be completed.
- iv. Apply for public status: A service provider will be able to apply for a trusted public status on the system. After requesting this status, the service provider will be required to complete a background check with a third-party provider. If the service provider passes the background check, they will be able to advertise their services to all users of the HomeTrumpeter system, rather than just the homeowner who invited them. As a service provider, I want to apply for public status so that I can advertise my services to all local homeowners on the HomeTrumpeter system.

#### 4.4 (Time Permitting) React Native Mobile App

If the team develops all functional requirements of the web app with time remaining, we will develop an Android/iOS app using React Native offering similar functionality to that offered by the web app.

#### 4.5 (Time Permitting) Machine Learning Model to Predict Service Provider Behavior

If the team develops all functional requirements of the web and React Native apps with time remaining, we will develop a machine learning model to evaluate the trustworthiness of a service provider. This model will use sentiment analysis on publicly available information related to the service provider such as news articles and social media posts to provide a prediction of behavior. This evaluation could serve

as part of the background check process for public service providers or be presented directly to homeowners considering their services.

\*\*

The team will be building against the 'Black Box' of HomeTrumpeter's API. Functionalities provided by the API are out of scope for the project and will not be developed or modified by team members. Little to no information regarding the inner workings of the API will be provided to the team. A conceptualization of assets, infrastructure, and threat agents that could be present in HomeTrumpeter's system will need to be created by the team for the purpose of threat modelling. Though we are not responsible for the development of the API, it is important to consider threats and assets that may be present in the black box to uphold the practice of developing a 'zero-trust' system.

## 5. Project Team and Roles

Collaboration within the development team has been facilitated through stand-up meetings, 3 times per week. Further communication has occurred through weekly/bi-weekly meetings with the stakeholders via Zoom. Additional weekly check-in meetings have taken place with the information system director to report progress. The team has been using Microsoft Teams to communicate amongst themselves. Teams was chosen due to its integration with our existing University of Winnipeg accounts, its file-sharing features, and video conferencing capabilities. Communication between the team and stakeholders has taken place primarily through email and Slack, due to difficulties adding external guests to a University of Winnipeg Teams group.

NOTE: All student members of the team will serve as developers in addition to the responsibilities included in their role.

The project roles and their respective responsibilities are as follows:

- **Scrum Master / Team Leader: Liam Kristjanson**
  - Lead the team in day-to-day organization and communication
  - Research the best practices for Agile development, and ensure the team follows these principles
  - Facilitate communication between the development team and stakeholders by acting as a primary contact point
- **Product Owner: Liam Kristjanson**
  - Create, maintain, and update the product backlog
  - Define a vision and direction for the project



- Work closely with stakeholders to prioritize and define requirements, ensure the product meets the business needs
- **Technical Leader: Xiao Zhang**
  - Coordinate technical requirements between the stakeholders and project team
  - Evaluate and select technologies that could be used to implement functional requirements in the product
- **Lead Developer: Vi Le**
  - Lead the team in the development process
  - Develop and enforce programming standards
- **Lead Quality Assurance Engineer: Arshjot Ghuman**
  - Lead the testing and quality assurance efforts of the team
  - Develop and maintain the CI/CD pipeline
  - Enforce testing coverage of the codebase
- **Lead Systems Analyst / Designer: Usmaan Sahar**
  - Gather and clarify requirements from stakeholders and users
  - Ensure the product conforms to usability goals and design principles
  - Lead the architecture and interface design efforts of the team
- **DevSecOps Engineer: W A Shadman**
  - Implement automated security checks within the continuous development and deployment of the product
  - Lead the overall product and process security efforts of the team
  - Identify and address potential vulnerabilities throughout the design and development processes
  - Identify, measure, and mitigate risks to the product and project
- **Agile Coach: Victor Balogun**
  - Oversee multiple agile teams within the organization
  - Provide mentorship for implementing agile principles
  - Provide training pertaining to agile principles
- **Industrial Mentor: Gabriel O**
  - Provide guidance relating to industrial technologies and practices
  - Provide training materials relating to required technologies
- **Stakeholders: HomeTrumpeter Team**
  - Provide requirements of the system
  - Provide pre-existing technical infrastructure (API, Private cloud)
  - Test and provide feedback on product releases

## 6. Timeline and Sprints

### **Sprint 1: Infrastructure and Conceptualization**

**(Oct 5 - 12)**

- Developed product backlog
- Created initial project plan.
- Created conceptual wireframes
- Created initial CI/CD pipeline infrastructure.
- Studied training materials related to required technologies

### **Sprint 2: Login with API Gateway I**

**(Oct 12 - 26)**

- Developed CI/CD pipeline to allow committed code to be automatically containerized and deployed to HomeTrumpeter clusters.
- Developed a functional prototype to demonstrate workflows such as creating an account, adding a property to the platform, inviting a tenant, and creating a service request.
- Developed pages against the HomeTrumpeter API allowing a user to create an account and log in to the system.
- Completed public deployment of initial web application.

### **Sprint 3: Login with API Gateway II**

**(Oct 26 - Nov 9)**

- Continued to develop CI/CD pipeline to include static code analysis and testing tools such as Lint and Jest.
- Defined a workflow for the development process, including a branching strategy, and promotion of environments for new code.
- Developed pages facilitating the rest of the account creation workflow including account role selection and phone verification.
- Developed a skeleton of the central dashboard that will serve as the atrium of our system.
- Performed threat modeling of our proposed system and HomeTrumpeter's existing API to identify assets, attack surfaces, and agents that could pose a threat to our app or HomeTrumpeter's API.
- Prepared for System Study Review

### **Sprint 4: Tenant Invitation I**

**(Nov 9 - 23)**

- Develop functionality allowing a homeowner or property manager to add their properties to the system.
- Develop functionality allowing a homeowner or property manager to invite tenants on the properties they have added to the system.
- Develop and implement penetration testing plan.

**Sprint 5: Tenant Invitation II**

**(Nov 23 - Dec 7)**

- Develop functionality allowing a tenant to accept an invitation from their landlord and create an account on the system.
- Develop the dashboard page to display information relevant to tenants including the property they are renting and their landlord's contact information.

**Sprint 6: Service Provider Onboarding I**

**(Dec 7 - 21)**

- Develop functionality allowing homeowners and property managers to invite their known service providers to the platform.
- Develop functionality allowing service providers to accept an invitation to the platform, create their own account and log in.

**Sprint 7: Service Provider Onboarding II**

**(Dec 21 - Jan 4)**

- Develop the dashboard page to display information relevant to service providers including quote requests, accepted jobs, and their advertised services.
- Develop functionality allowing service providers to add services to their account such as HVAC, plumbing, and carpentry according to their training.
- Prepare for and present Detailed Design Review

**Sprint 8: Service Requests I**

**(Jan 4 - 18)**

- Develop functionality allowing a tenant to create a service request that will be sent to their landlord.
- Develop functionality allowing a homeowner or property manager to browse service providers and request a quote on receipt of a service request.

- Develop functionality allowing a service provider to send a quote and proposed date to a homeowner or landlord in response to a request for quote.

**Sprint 9: Service Requests II**

**(Jan 18 - Feb 1)**

- Develop functionality allowing a homeowner to approve a quote from a service provider and notify the provider of the approval.
- Develop functionality allowing the homeowner or property manager to rate the services of the service provider after completion of a maintenance request.

**Sprint 10: Background Check for Public Status I**

**(Feb 1 - 15)**

- Develop functionality allowing a service provider to apply for public status, sending a request for a background check for third-party provider Certn.
- Develop functionality to process completed background checks and grant public status to service providers who pass.

**Sprint 11: Background Check for Public Status II**

**(Feb 15 - 29)**

- Develop functionality allowing service providers with public status to offer their services to all homeowners and property managers on the platform.
- Begin development of React Native mobile frontend, if time allows.

**Sprint 12: Service Provider ML Model / React Native Application**

**(Feb 29 - Mar 14)**

- Continue development of React Native mobile frontend
- Research feasibility of a machine learning model to predict service provider behavior based on public information.
- Begin development of machine learning model to predict service provider behavior.
- Determine how machine learning model could be integrated into the system.
- Prepare for Project Completion Report

**Sprint 13: Service Provider ML Model / React Native Application**

**(Mar 14 - 21)**

- Continue development of React Native mobile frontend.

- Integrate machine learning model for service provider behavior into the system.
- Conduct regression testing and training of machine learning model for service provider behavior.
- Present project completion report.

## 7. Product Backlog

The product backlog was derived from a list of high level of requirements provided by the sponsor, as well as a review of the functionalities offered by the current HomeTrumpeter system. A set of user stories was created based on the target user groups and list of requirements. These user stories were then distributed between a set of two-week sprints spanning from the start date to the project deadline. This set of user stories forms the product backlog and represents the functionalities we plan to have implemented in the system at the end of each sprint.

Before each sprint, each of the user stories involved in the sprint are broken down into individual work items and assigned to team members. Breaking apart user stories into individual tasks iteratively rather than at the beginning of the project allows us to create tasks with a better understanding of the dependencies, constraints, and effort required based on our previous development of the system. Any change in requirements can be accommodated by inserting the necessary change into the backlog at an appropriate level of priority. This framework allows the team to adapt to change and respond to unknown variables by moving backlog items earlier if the team is working ahead of time, or later if work is behind schedule.

Requirements were prioritized based on their dependencies to one another. For example, login and account creation functionalities needed to be implemented before work on property management and tenant invitation functionalities began. The importance of each requirement and user story to the functional viability of the system were also weighed in evaluating their priority within the backlog. Functional requirements of the system such as service request creation were prioritized ahead of 'Nice to have' features such as a React Native mobile app, or machine learning model.

The current state of the product backlog is given in the appendix. Note how completed and in-progress work items have been broken down into individual tasks, and requirements in later sprints are left as high-level user stories.

## 8. Sprint Planning and Execution

Sprint planning meetings were held every two weeks, involving the development team and stakeholders. These meetings allowed the team the opportunity to present our proposed objectives for the upcoming sprint, and the work items necessary to complete these goals. Stakeholders would then be able to offer input on the planned objectives, which could be adjusted accordingly. Each work item was assigned a Story point estimation' according to the estimated amount of effort required to complete the work item. Work items were assigned to sprints based on the number of story points the team had historically been able to complete in a two-week sprint. Each high-level requirement was allotted two sprints to be completed. If we complete a requirement ahead of schedule, we will be able to pull backlog items from later sprints into the current sprint backlog to work ahead.

In the planning meeting preceding Sprint 3, stakeholders expressed a desire for the CI/CD infrastructure to be prioritized ahead of some features the team had planned to develop during the sprint. A feature initially planned for Sprint 3 was placed into the backlog for a later sprint to accommodate the additional work required to further develop the CI/CD infrastructure. These sprint planning meetings along with the sprint backlog structure allows such collaborative planning to be possible.

## 9. Daily Standups

Our team conducts standup meetings three times a week, on Monday, Wednesday, and Friday at 1:00PM. These are short, informal meetings where each team member shares what they have worked on the previous day, what they plan to work on the next day, and any issues blocking their progress. These meetings encourage accountability, as each team member needs to share how much progress they have made. Our team meets face-to-face or via video call on Mondays and Wednesdays. Due to scheduling conflicts, our team has adopted an asynchronous scrum on Fridays. In the asynchronous scrum, each team member is expected to send a message regarding their progress whenever they can.

This standup schedule balances the benefits of meeting face-to-face regularly with the challenges of coordinating six students with six different schedules. As the agile methodology emphasizes continuous process improvement, our scrum format may be subject to change. If our team feels the need to increase or decrease meeting frequency to stay on track, such a change could be proposed at the sprint retrospective meetings.

## 10. Continuous Integration and Testing

We have integrated continuous integration (CI) and continuous deployment (CD) into our development workflow using the GitLab CI/CD pipeline. This powerful tool streamlines various tasks, including building our product and deploying it onto the OpenShift platform. Here's how our pipeline operates:

**Feature Development and Bug Fixes:** When our developers work on new features or bug fixes, they branch out from the main repository. After coding is completed, they push their changes from their local environments to the GitLab repository.

**Automated Building and Testing:** GitLab automatically triggers the build process and runs a battery of tests. We have an array of testing processes that ensure the code quality, and we'll dive deeper into these tests shortly.

**Test Failure Handling:** If the code fails any of the tests, the developer receives feedback and is required to address the issues promptly.

**Merging to Development Branch:** Once the code is fixed and passes the tests, it is pushed back to be merged into the 'develop' branch. This branch acts as an intermediary staging area before deployment to the main branch.

**Develop Branch Verification:** When the code is merged into the 'develop' branch, it undergoes another round of building and testing (double-checking) to ensure its stability. An image is then released for deployment.

**Deployment:** The deployment to OpenShift is managed by a dedicated 'deploy' job. This deployment is distinct from the live deployment and allows Quality Assurance (QA) teams and other stakeholders to review the software to ensure it functions as expected.

**Main Branch Update:** Following the successful review of the deployment in the 'develop' branch, it is merged into the 'main' branch. This process implements a Rolling Update strategy, seamlessly replacing the previous public deployment with the new one.

**Testing Strategy:** Our testing strategy is multifaceted, encompassing various types of tests to ensure the robustness and reliability of our product. Here's a breakdown:

**Unit Testing:** We employ Jest and React Testing Library for basic unit testing. These tests validate individual components, functions, or units of code to ensure they work correctly. We are transitioning towards a Test-Driven Development (TDD) approach where our QA team writes tests for functional components before the development team writes the corresponding code. This process ensures that code changes are verified against expected behavior.

**Integration Testing:** Integration testing verifies the interactions and compatibility between different components of our application. It ensures that various parts of the system work harmoniously together, simulating real-world usage scenarios.

**Regression Testing:** Regression testing is crucial in CI/CD pipelines. It involves re-running previously executed tests to detect and prevent the introduction of new bugs or issues as code evolves. It helps maintain the stability of our application.

**End-to-End (E2E) Testing:** For E2E testing, we use Selenium. E2E tests simulate user interactions with the application, checking the entire flow from start to finish. This testing approach ensures that all components of the system work together as expected.

As we continue to improve our testing practices and tooling, we are gradually reducing manual testing, as was the case in Sprint 1. We recognize the importance of automated testing to ensure a smooth and efficient development process and are actively learning how to leverage these tools and technologies to their fullest potential. Our goal is to shift towards a fully automated and comprehensive testing strategy, reducing the reliance on manual testing.

## 11. Retrospectives and Continuous Improvement

At the conclusion of each 2-week sprint, the team holds a sprint retrospective involving just the development team. The purpose of the retrospective will be to share what worked well in the sprint, discover processes or interactions that could be improved, and to commit to concrete plans of action according to these findings.

Each member of the team will be strongly encouraged to share a positive aspect of the previous sprint, and an area of improvement. The team will then be able to find common themes among positive experiences and identify pain points that could be removed. This will allow each team member to have a say in the process changes enacted in the following sprint. The Scrum Master will be responsible for organizing and facilitating this meeting. This candid, self-organizing, regularly occurring retrospective will facilitate the continuous learning and improvement sought by agile teams.

A positive change that came from the sprint retrospective meetings was the development of a standardized development workflow. At the conclusion of Sprint 2, our team agreed that a standardized process for branching from the repository and merging new code should be created. The objective of this change was to eliminate confusion regarding the purpose of each feature branch and ensure code changes are made in a safe and consistent manner. Such areas of improvement will continue to be found and be acted upon as we carry out the development process.



## 12. Risk Management

Identifying potential risks and challenges and outlining a plan to monitor, assess, and mitigate them is a crucial component in the modern software development approach. The following are some of the risks that have been identified, along with their corresponding monitoring and assessment techniques and mitigation plans that we will implement:

Risk	Monitoring	Assessment	Mitigation
Technical Challenges	Monitor development progress and track integrated system performance	Review technical task status and assess task completion against schedule	Schedule extra time for troubleshooting, engage HomeTrumpeter's technical team, conduct thorough testing
Scope Creep	Monitor project progress and review product backlog	Assess requirement changes and review stakeholder feature requests	Implement formal change request process, prioritize new requests for future sprints
Resource Constraints	Monitor team workloads and track external dependencies	Evaluate resource impact on timelines	Manage commitments, reallocate tasks as needed, communicate with stakeholders about external delays
Security Vulnerabilities	Conduct security assessments and review security measures	Identify and evaluate security vulnerabilities	Perform regular security testing and code reviews, use automated security checks in CI/CD, penetration testing, implement strong authentication measures

External Dependencies	Track availability and performance of external services	Assess impact of external service disruptions or changes	Develop contingency plans, implement fallback mechanisms, maintain communication with third-party providers
Scope Uncertainty	Seek clarification from HomeTrumpeter and stakeholders	Evaluate gaps in stakeholder requirements	Maintain open communication with HomeTrumpeter, hold regular requirement clarification meetings, document all requirements
Time Management	Monitor sprint progress and review project timeline	Assess task completion within timeframes	Adjust sprint planning, allocate additional time as needed, refine sprint planning process

A key concern in this model of risks is the potential for security vulnerabilities in the delivered product. To monitor potential threat agents, attack surfaces, and find appropriate countermeasures for each, our team has constructed a threat model for the proposed and existing systems. A full copy of our essential assets, abuse cases, and potential threat agents can be found in this threat model in the appendix. This threat model serves as the basis for the security measures taken in this project. As functionality continues to be added, potential attack surfaces will emerge. As such, the threat model will continually evolve as the project progresses.

## 13. Budget and Resources

### 13.1 Allocation of Budget and Resources

The team has a weekly total of 72 person-hours of resources available based on the University of Winnipeg definition of a 6 credit hour course. This includes 3 hours of weekly lecture material and 9 hours of weekly studying.

$$\begin{aligned}
 \text{Weekly Resources} &= (3 \text{ hours/week} + 9 \text{ hours/week}) * 6 \text{ persons} \\
 &= 72 \text{ person-hours/week}
 \end{aligned}$$

We have assigned 'story points' to each item in the sprints. Story points are a comparative measure of the estimated amount of effort required to complete a given task. As development continues, we will better understand the number of story-points the team can complete in a two-week sprint. In sprints 2 and 3, our team was able to complete an average of 33 story points. In future sprint planning meetings, we will create a sprint backlog of items worth roughly 33 total story points. This will allow us to commit to an amount of work that is consistent and achievable each sprint.

We track the overall status of the sprint through a burndown chart. The burndown chart compares the number of story points remaining to the number of days remaining in a sprint. If the burndown chart demonstrates that the sprint is behind schedule at day 7, the team will have to take action to ensure the goals of the sprint can still be met. Such courses of action could include allocation of additional person-hours to the sprint, removal of low-priority tasks from the sprint backlog, or a change in the structure of some tasks to improve efficiency. Examples of our burndown charts for sprints 2 and 3 can be found in the appendix. Later sprints are currently planned for significantly less than 33 story points to allow for overhead to fulfill additional requests and requirements identified in stakeholder feedback.

### **13.2 Tool Selection**

All tools that have been selected for the development process are free or provided by the University of Winnipeg and HomeTrumpeter.

The collaboration tools selected are Microsoft Teams, Slack, and Zoom. Microsoft Teams is connected to University of Winnipeg Outlook accounts, which allows easy collaboration amongst faculty and team members on documents and slideshows, as well as receiving tech support by the University of Winnipeg IT Department. Slack and Zoom were chosen as communication platforms with HomeTrumpeter members due to their familiarity with the platforms and reliability.

The project is managed using JIRA, which includes various Agile tools such as issue tracking, product backlog tracking, and development branch tracking. As JIRA is the most-used project management software, Atlassian maintains a robust infrastructure to ensure high availability for its cloud services.

Infrastructure was mostly provided by HomeTrumpeter. The product is deployed to the HomeTrumpeter's OpenShift private cloud platform, which has extensive support for CI/CD pipelines and container orchestration. The project is hosted in a GitLab repository on HomeTrumpeter's private cloud. GitLab greatly benefits the Agile development process as it offers a built-in CI/CD pipeline, as well as numerous integrations with various IDEs and project management tools (e.g., Jira).

The development framework chosen is React JS and React Native. Both use a component-based architecture, which promotes reusability of complex code and saves resources. The similar architecture between the two allows for ease of migration as most of the logic remains the same despite the difference in WebView and native mobile components. React is highly reliable as it is maintained by Meta and has the largest community and repository of third-party libraries, which may be used to enhance the quality and rate of development. Additionally, the project will use TypeScript as the programming language, Vite as the development, and Rollup to bundle modules for production. TypeScript serves to promote consistent coding standards as all types are checked at runtime, which prevents type errors from entering production and aids in unit testing. Vite is chosen as the development environment to provide hot reload, quick compilation, and its use of Rollup for building the codebase. Rollup optimizes all static assets to be delivered efficiently in production.

All tools chosen for Prove IT supports rapid development, stakeholder participation, and high reliability, to promote an Agile development process.

## 14. Benefits and Expected Outcomes

Through the development of this project, HomeTrumpeter will be able to evaluate the viability of constructing an external application leveraging the API of their flagship system. This will allow HomeTrumpeter to identify strengths and weaknesses of their API as they look to enter the ever-expanding API economy. The student development team stands to benefit greatly from the project as well. Members of the development team will gain valuable experience using industry-standard technologies. An inside experience of the entire software development lifecycle will also be invaluable. The agile process has been adopted by most organizations in the industry, and it is a natural fit for the needs of this project.

The continuous learning emphasized by the agile approach will allow the team to learn valuable lessons through hands-on experience. Rather than attempting to plan for every requirement upfront, agile emphasizes iterative development and responsiveness to change. These qualities will prove paramount to the success of this project, as taking on a project for the first time as a team is bound to bring with it a high level of change and uncertainty.

The iterative process of planning, development, and review will increase product quality and learning opportunities for the team. By focusing on delivering functional software with each iteration, missing requirements, bugs, and test failures can be addressed much sooner than with the waterfall approach. Beginning the development process as early as possible allows stakeholders to interact with a functional product very early in the development process. This will allow stakeholders with industrial

experience to offer feedback on the product at more intervals throughout the project, creating more learning opportunities for the development team. The agile approach will do well to facilitate the level of collaboration, quality, and rapid product delivery that will prove paramount to the success of this project.

## 15. Design Principles and Expectations

During the design process many factors were considered. The primary inspiration was maintaining a theme with the current design of HomeTrumpeter's services while also ensuring that we add our own twist and flair to it. When we initially produced the wireframes, our design was very similar to the current design HomeTrumpeter uses. But after some discussion with our sponsors and continuous support for innovative ideas, the choice was made to go for a more simplistic yet functional design. We intended to create multiple pages for each service rather than what the current design holds which creates a tab on the same page over the previous. This was done to ensure that users were not overwhelmed and had a clear idea on what was currently being accessed or which service was being used. Another principle that was incorporated was the idea that each screen a user is met with contains all the information they need without it being overbearing. Each aspect of the screen should be clear and concise so that the user is easily able to comprehend exactly what they are requesting from the platform.

Other aspects are still being developed. Currently there is still more work to be done on the consistency of our designs across the multiple pages as the development across each has been incorporating modern design ideas for each one. Another aspect is to ensure that the clarity we are going for with each page is properly met. In its current state, the design between a tenet's dashboard and a homeowner's dashboard are the same even though homeowner is having more features and responsibilities than tenets. The process for service providers is still being developed and considered and errors in the design are being searched for within the prototypes to ensure that when implication occurs it is done as smoothly as possible.

Current design feedback from the sponsors also goes into consideration for the future designs. Currently the sponsors have shown a high approval of the current design if not for a few criticisms here and there. However, as the project progresses there will be more high functioning features that require a more detailed inspection on whether the design is appropriate or not. The methods of searching and understanding the design requirements are in a constant effect and it is a challenging but worthwhile experience to create the platform with everything accounted for.

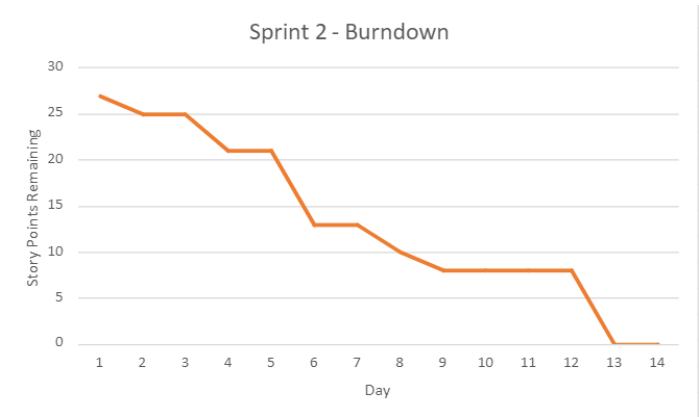
# 16. Appendix

## 16.1 Product Backlog and Work Breakdown Structure


	Task Number	Task Name	Resource	Duration	Story Points	Start	Finish	Status
2	1.0	User Stories			11			
3	1.1	Identify Key Stakeholders	Project Team		2	06 Sep 2023 1:00PM	06 Sep 2023 2:15PM	Done
4	1.2	Form Project Team	Project Manager		1	06 Sep 2023 1:00PM	06 Sep 2023 2:15PM	Done
5	1.3	Project Proposal	Project Team		8	06 Sep 2023 1:00PM	25 Sep 2023 11:59PM	Done
6	2.0	Product Backlog			4			
7	2.1	Create Product Backlog	Product Owner		3	13 Sept 2023 1:00PM	30 Sept 2023 12:00PM	Done
8	2.2	Assign Story Point Estimations	Product Owner		1	27 Sept 2023 1:00PM	1 Oct 2023 12:00PM	Done
9	3.0	High Level Sprint Planning			3			
10	3.1	Create Project Plan	Project Manager		3	27 Sept 2023 1:00PM	06 Oct 2023 11:59PM	Done
11	3.2	Approve Project Plan	Stakeholders		0			Done
12	4.0	Sprint - 1 - Infrastructure and Learning			4			
13	4.1	Sprint 1 - Planning Meeting	ALL		0	05 Oct 2023 6:00PM	05 Oct 2023 7:00PM	Done
14	4.3	Login Gateway Wireframe	Usmaan		2	05 Oct 2023 7:00PM	12 Oct 2023 10:00AM	Done
15	4.4	Add Property Wireframe	Usmaan		2	05 Oct 2023 7:00PM	12 Oct 2023 10:00AM	Done
16	5.0	Sprint - 2 - Login with API Portal I			27			
17	5.1	Sprint 2 - Planning / Sprint 1 Demo	ALL		0	12 Oct 2023 6:00PM	12 Oct 2023 7:00PM	Done
18	5.2	Develop CI/CD Pipeline - Simple Deployment	QA Engineer		3	05 Oct 2023 7:00PM	16 Oct 2023 7:30PM	Done
19	5.3	Homeowner/Manager Create Account	Lead Programmer		8	18 Oct 2023 12:00PM	26 Oct 2023 12:00PM	Done
20	5.4	Homeowner/Manager Login	Technical Lead		5	12 Oct 2023 6:00PM	18 Oct 2023 12:00PM	Done
21	5.6	Prototype - Create Account/Login	Product Owner		1	12 Oct 2023 6:00PM	13 Oct 2023 12:00PM	Done
22	5.7	Prototype - Add Property	Product Owner		1	13 Oct 2023 12:00PM	13 Oct 2023 6:00PM	Done
23	5.8	Prototype - Invite Tenant	Product Owner		1	13 Oct 2023 6:00PM	16 Oct 2023 6:00PM	Done
24	5.9	Prototype - Invite Service Provider	Lead Designer		1	12 Oct 2023 6:00PM	21 Oct 2023 12:00PM	Done
25	5.10	Prototype - Create Service Request	Lead Designer		1	12 Oct 2023 6:00PM	21 Oct 2023 12:00PM	Done
26	5.11	Research Testing Requirements and Tools	DevSecOps		3	16 Oct 2023 12:00PM	20 Oct 2023 12:00PM	Done
27	5.12	Research and Select Tech Stack	Technical Lead		3	12 Oct 2023 6:00PM	18 Oct 2023 2:00PM	Done
	Task Number	Task Name	Resource	Duration	Story Points	Start	Finish	Status
	6.0	Sprint - 3 - Login with API Portal II			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		In Progress
	6.5	CI/CD Pipeline - Implement Security Scans	QA Engineer/DevSecOps		3	26 Oct 2023 7:30PM	31 Oct 2023 12:0PM	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
	6.1	Homeowner - Verify Phone Number	Project Lead		2	26 Oct 2023 7:30PM	1 Nov 2023 12:30PM	Done
	6.1	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			In Progress
	7.0	Sprint - 4 - Tenant Onboarding I			38			
	7.1	Sprint 4 - Planning / Sprint 3 Demo	ALL		0	09 Nov 2023 6:30PM	09 Nov 2023 7:30PM	Not Started
	7.2	Homeowner - Add Property	Lead Developer		8			Not Started
	7.3	Dashboard - View Properties	Technical Lead		3			Not Started
	7.4	Fix Missing XCK in Production Deployment	Unassigned		3			Not Started
	7.5	Protect Routes for SetRole, Phone Verification	Unassigned		3			Not Started
	7.6	Dashboard - Delete Property	Unassigned		1			Not Started
	7.7	Tenant Invitation Wireframe	Unassigned		2			Not Started
	7.8	Send Tenant Invitation	Unassigned		5			Not Started
	7.9	System Study Review	Project Team		13	13 Nov 2023 1:00PM	13 Nov 2023 2:15PM	Not Started
	7.10	Implement Penetration Testing Plan	DevSecOps/QA Engineer		5			Not Started

12.0	Sprint - 9 - Service Request Creation II			21			
12.1	Sprint 9 - Planning / Sprint 8 Demo	ALL		0	18 Jan 2024 6:30PM	18 Jan 2024 7:30PM	Not Started
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
12.8	Service Request - Testing	Unassigned		8			Not Started
13.0	Sprint - 10 - Background Check I			26			
13.1	Sprint 10 - Planning / Sprint 9 Demo	ALL		0	01 Feb 2024 6:30PM	01 Feb 2024 7:30PM	Not Started
13.2	Background Check Wireframe	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
14.0	Sprint - 11 - Background Check II			16			
14.1	Sprint 11 - Planning / Sprint 10 Demo	ALL		0	15 Feb 2024 6:30PM	15 Feb 2024 7:30PM	Not Started
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
15.0	Sprint - 12 - Service Provider ML Model			19			
15.1	Sprint 12 - Planning / Sprint 11 Demo	ALL		0	29 Feb 2024 6:30PM	29 Feb 2024 7:30PM	Not Started
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
16.0	Sprint - 13 - Service Provider ML Model			24			
16.1	Sprint 13 - Planning / Sprint 12 Demo	ALL		0	14 Mar 2024 6:30PM	14 Mar 2024 7:30PM	Not Started
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

16.2 Burndown Charts



16.3 User Interface Mock-ups (From Figma Prototype)




Welcome back to

Log in

[Forgot password?](#)

Don't have an account? [Register](#)



Enter the four-digit code sent to  
(204) 222-\*\*\*\*

1

2

3

4

Verify

Didn't get a text? [Re-send SMS](#)





## Add a Property

Property info

Street Number\*

221

Street Name\*

Hill Street

Unit Number

City\*

Milwaukee

State\*

WI

ZIP Code\*

53021

Estimated Rent

\$1800

Registration Fee

\$50

Add Property



## Dashboard



### Properties

221 Hill Street, Milwaukee, WI

Add a property...



### Service Requests

Service	Provider	Property
Request a service		

### Tenants

Name	Property	Status
Bob Ross	221 Hill Street	Invite Sent
Add a tenant		

### Tips

221 Hill Street does not have a tenant. [Invite a tenant](#)




## Invite a Tenant

Invite your tenants so that they can make service requests at their rental properties

Tenant information

Property

221 Hill St. 

First Name

Bob

Last Name

Ross

E-Mail

bobross@happytrees.com

Phone

(204) 777-7777


Send Invite




## Dashboard



### Properties

221 Hill Street, Milwaukee, WI	 
--------------------------------	---

### Service Requests

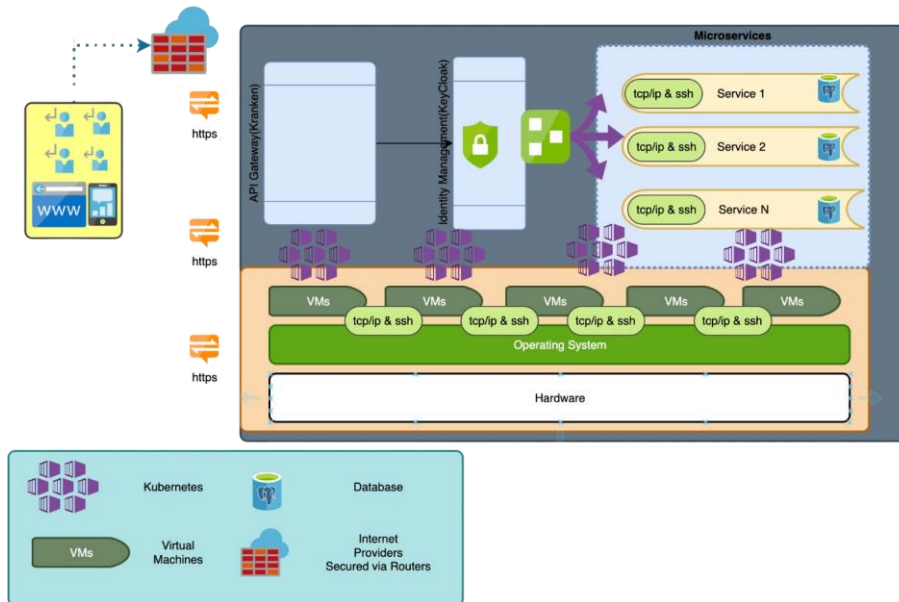
Service	Provider	Property
Sink and shower in...	Plumbing service	221 Hill Stre... 
<a href="#">Request a service</a>		

### Tips

16.4 Abuse Cases and Countermeasures

Use Case #	Asset	Abuse Use Case	Countermeasures
1	Web Application Database	Unauthorized Access to Homeowner and Tenant Data	Implement strong authentication mechanisms, input validation, parameterized queries, sanitize user inputs, use a Web Application Firewall (WAF), conduct regular security audits, ensure strong database authentication, encrypt sensitive data in transit and at rest, regularly patch the web application.
2	Communication Channel	Man-in-the-Middle Attack on Communication Channels	Use strong encryption protocols for data in transit, implement HTTPS for all communications.
3	API Gateway	API Token Theft via API Vulnerabilities	Implement API rate limiting, use input validation to prevent injection attacks, regularly update and patch the API.
4	Container Infrastructure	Malicious Activities in Containers	Use trusted base images, regularly scan for vulnerabilities, employ runtime security for containers.
5	Kubernetes Configuration	Misconfigurations in Kubernetes	Ensure Kubernetes configurations are secure, regularly update Kubernetes, monitor for vulnerabilities.
6	API Gateway	DDoS Attack on API Gateway	Implement rate limiting, use services like Cloudflare to mitigate DDoS attacks.
7	Identity Management System	Unauthorized Session Access via Identity Management Weaknesses	Use multi-factor authentication, implement session management best practices, auto logout after inactivity.
8	Database	Database Tampering	Encrypt sensitive data in the database, implement strong authentication mechanisms for database access, regularly monitor and audit database activities.

## 16.5 Architectural Diagram and Threat Model



Note: Items underlined are assumed to be conducted by HT.

### Assets:

1. **Homeowner Data**: Personal details, contact information, property ownership records.
2. **Tenant Data**: Personal details, contact information, rental history.
3. **Property Details**: Location, size, amenities, price, availability status.
4. **Tickets and Tasks Details**: Maintenance requests, complaints, service requests.
5. **Payment Details and History**: Transaction records, bank details, credit card information.
6. **Credentials**: API keys, SSH keys, tokens, and any other credentials used for authentication and authorization.
7. **Sensitive Application Data**: business logic, process data, application settings, environment variables etc.
8. **Infrastructure**: physical hardware, hosted virtual machines and operating systems, Kubernetes cluster, routers, switches, load balancers and firewalls.
9. **Applications**: The web and mobile applications, microservices, API Gateway (Kranken), and Identity Management (KeyCloak).

### Threat Agents:

1. **Unauthorized External Users:** Individuals outside the organization trying to gain unauthorized access.
2. **Unauthorized Internal Users:** Employees or associates without the necessary permissions trying to access restricted data.
3. **Authorized Homeowners, Tenants, and Service Providers with Malicious Intent:** Legitimate users who misuse their access rights.
4. **Man-in-the-Middle Attackers:** Attackers who intercept and possibly alter the communication between two parties without their knowledge.
5. **API Attackers:** Individuals targeting the API gateway or microservices directly.
6. **Viruses, Worms, Ransomware, etc.:** Malicious software designed to harm, exploit, or extract data.

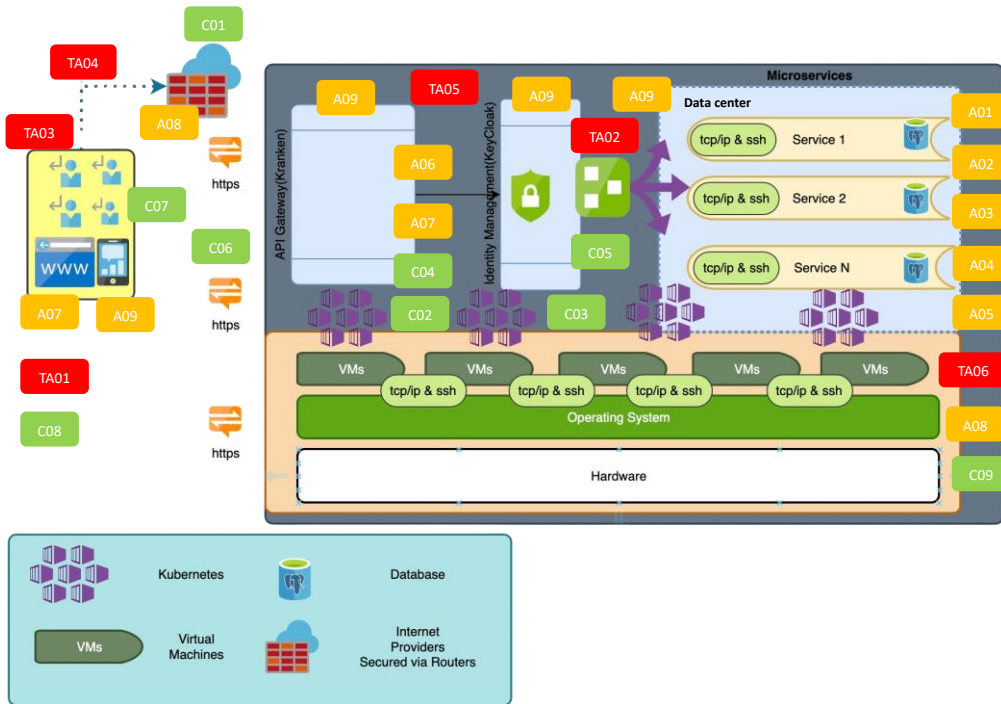
### Weaknesses/Vulnerabilities/Attack Surfaces:

1. **Network:** Inadequate network security could allow for lateral movement within the system.
2. **Containers:** If not properly secured, containers can be an entry point for attackers. Vulnerabilities in container images or misconfigurations can be exploited.
3. **Kubernetes Orchestration:** Misconfigurations in Kubernetes can expose sensitive information or control interfaces.
4. **API Gateway (Kranken):** If not properly secured, it can be a target for DDoS attacks, injection attacks, etc.
5. **Identity Management (KeyCloak):** Weaknesses in authentication or session management can lead to unauthorized access.
6. **Communication Channels:** Unencrypted or weakly encrypted communication can be intercepted and read by attackers.
7. **Web and Mobile Applications:** Vulnerabilities in the application code can be exploited to gain unauthorized access or execute malicious actions.
8. **Database:** If not properly secured, sensitive data can be extracted or tampered with.

### Controls:

1. **Network Security:** Firewalls, intrusion detection/prevention systems (IDS/IPS), and network segmentation.
2. **Container Security:** Use trusted base images, regularly scan for vulnerabilities, and employ runtime security.
3. **Kubernetes Security:** Limit access to the Kubernetes API, use network policies, and regularly audit configurations.
4. **API Security:** Implement rate limiting, use strong authentication and authorization mechanisms, and regularly audit and test for vulnerabilities.

5. **Identity Management:** Use multi-factor authentication, regularly review access rights, least privilege principles and use strong encryption for data at rest and in transit.
6. **Encryption:** Use strong encryption protocols for data in transit and at rest.
7. **Regular Patching:** Regularly update all software components to patch known vulnerabilities.
8. **Monitoring and Logging:** Continuously monitor the system for suspicious activities and maintain logs for forensic purposes.
9. **User Training:** Regular security awareness training for all users to recognize phishing attempts and other social engineering attacks.



### Assets

A01: Homeowner Data  
A02: Tenant Data  
A03: Property Details  
A04: Tickets and Tasks Details  
A05: Payment Details and History  
A06: Credentials  
A07: Sensitive Application Data  
A08: Infrastructure  
A09: Applications

### Threat Agents

TA01: Unauthorized External Users  
TA02: Unauthorized Internal Users  
TA03: Authorized Homeowners, Tenants, and Service Providers with Malicious Intent  
TA04: Man-in-the-Middle Attackers  
TA05: API Attackers  
TA06: Viruses, Worms, Ransomware, etc.

### Controls

C01: Network Security  
C02: Container Security  
C03: Kubernetes Security  
C04: API Security  
C05: Identity Management  
C06: Encryption  
C07: Regular Patching  
C08: Monitoring and Logging  
C09: User Training

16.6 Penetration Testing Plan and Implementation

Objective	Target System/Component	Tester	Tools/Methodologies Used	Expected Outcome
API Security Assessment	HomeTrumpeter API Gateway	Penetration Tester 1	OWASP API Security Top Ten, Manual Testing, API Scanners	Identify vulnerabilities in API endpoints, test for injection attacks, and assess access controls. Ensure API security compliance (by HT).
Authentication and Authorization Testing	User Authentication and Authorization System	Penetration Tester 2	Manual Testing, Burp Suite, OAuth testing tools	Test login, access controls, and session management. Identify weaknesses in authentication mechanisms. Verify proper authorization checks.
Database Security Assessment	Database Server and Data Storage	Penetration Tester 1	Database Scanners, Manual Testing, SQL Injection Testing	Test for SQL injection vulnerabilities, data exposure, and unauthorized access. Verify encryption and access controls on data storage.
Container Security Assessment	Containerized Application Deployment	Penetration Tester 3	Container Scanners, Manual Testing, Docker Security Tools	Assess container security by identifying vulnerabilities, misconfigurations, and weaknesses in container images.
Kubernetes Security Assessment	Kubernetes Orchestration Platform	Penetration Tester 2	Kubernetes Security Scanners, Manual Testing	Review Kubernetes configurations for security



				weaknesses and ensure proper access controls and network policies.
Web Application Security Testing	React Web Application	Penetration Tester 3	OWASP Top Ten, Burp Suite, Selenium, Manual Testing	Identify web application vulnerabilities such as XSS, CSRF, and other common web application security issues.
End-to-End Testing	Entire System Workflow	Penetration Tester 1, 2, 3	Manual Testing, Automation Testing	Verify the security of the end-to-end workflow, ensuring that all components interact securely and that vulnerabilities are not introduced as data flows through the system.
Threat Modeling Assessment	Entire System Architecture	Penetration Tester 1, 2, 3	Threat Modeling Frameworks, Manual Assessment	Conduct a threat modeling exercise to identify potential threats, assets, and vulnerabilities in the system architecture. Develop recommendations for mitigating identified threats.
External Dependency Assessment	Third-party Background Check Provider	Penetration Tester 1	Manual Testing, Communication Assessment	Assess the security of interactions with external third-party services. Verify proper authentication and encryption when handling sensitive data.

Continuous Monitoring Assessment	Ongoing System Health and Security	Penetration Tester 2	Automated Monitoring Tools, Log Analysis	Review ongoing system logs and alerts to detect any unusual or unauthorized activities. Ensure continuous monitoring and response mechanisms are effective.
----------------------------------	------------------------------------	----------------------	--	---

**Note:** The penetration testing will be implemented in the later sprints as suggested by HomeTrumpeter. The testing team will consist of three testers with expertise in different areas to cover various aspects of the system's security. Each tester will perform their assessments independently, and the findings will be consolidated into a comprehensive security assessment report. The expected outcome of each test is to identify vulnerabilities, weaknesses, or misconfigurations and provide recommendations for remediation.