

An Exploration of Turing Pi Based Edge Cloud with Docker/Kubernetes

DESIGN DOCUMENT

Team: SDMAY23-19

Client: Dr. Akhilesh Tyagi

Advisor: Dr. Nicholas Fila

Team Members:

Chee Hau Fong

Theodore Thayib

Bryce Carter

James Gibbs

Kyoungkeun Lee

Derrick Brandt

sdmay23-19@iastate.edu

<https://sdmay23-19.sd.ece.iastate.edu/>

Revised: 12/2/2022

Executive Summary

Development Standards & Practices Used

Agile development practices

Summary of Requirements

- The senior design product shall implement a scalable message service.
- While the Turing Pi is online, the senior design product shall use an open source distributed event streaming platform.
- The senior design project shall have three compute modules available at any given time.
- The service landing page should maintain smooth performance scalable to 30,000 requests per second.
- The messaging service shall propagate messages to user groups within .75 seconds.
- If the messaging service is under 90% load or greater, then the service shall propagate messages to user groups within 2 seconds.
- The service shall be available to United States users 98.99% of the month.
- The messaging service shall encrypt user messages in transit.
- The senior design product shall use the Hypriot operating system.
- The senior design product shall use Docker as its containerization platform.
- The senior design product shall use Kubernetes as its orchestration platform.
- The senior design product shall use CI/CD pipelines for building and testing code.
- The turing pi product shall be placed inside a mini ITX case.
- The senior design product shall use a minimum of three raspberry pi compute modules.
- The senior design product shall implement a simple graphical user interface.

Applicable Courses from Iowa State University Curriculum

- | | | |
|-------------|------------|----------|
| • CPR E 288 | • ComS 309 | • SE 317 |
| • CPR E 319 | • CS 252 | • SE 319 |

New Skills/Knowledge acquired that was not taught in courses

- Turing Pi cluster board
- Kubernetes
- Docker
- Kafka

Table of Contents

1 Team	5
2 Introduction	5
3 Project Plan	8
3.1 Project Management/Tracking Procedures	8
3.2 Task Decomposition	8
3.3 Project Proposed Milestones, Metrics, and Evaluation Criteria	8
3.4 Project Timeline/Schedule	9
3.5 Risks And Risk Management/Mitigation	9
3.6 Personnel Effort Requirements	10
3.7 Other Resource Requirements	10
4 Design	10
4.1 Design Context	10
4.1.1 Broader Context	10
4.1.2 Prior Work/Solutions	11
4.1.3 Technical Complexity	11
4.2 Design Exploration	12
4.2.1 Design Decisions	12
4.2.2 Ideation	12
4.2.3 Decision-Making and Trade-Off	12
4.3 Proposed Design	13
4.3.1 Overview	13
4.3.2 Detailed Design and Visual(s)	14
Web Application Side:	16
4.3.3 Functionality	17
4.3.4 Areas of Concern and Development	17
4.4 Technology Considerations	17
4.5 Design Analysis	17

5	Testing	17
5.1	Unit Testing	17
5.2	Interface Testing	18
5.3	Integration Testing	18
5.4	System Testing	18
5.5	Regression Testing	18
5.6	Acceptance Testing	18
5.7	Security Testing (if applicable)	18
5.8	Results	18
6	Implementation	18
7	Professional Responsibility	18
7.1	Areas of Responsibility	19
7.2	Project Specific Professional Responsibility Areas	20
7.3	Most Applicable Professional Responsibility Area	20
8	Closing Material	20
8.1	Discussion	20
8.2	Conclusion	21
8.3	References	21
8.4.1	Team Contract	21
	Team Procedures	21
	Participation Expectations	22
	Leadership	23
	Collaboration and Inclusion	23
	Goal-Setting, Planning, and Execution	24
	Consequences for Not Adhering to Team Contract	25

List of figures/tables/symbols/definitions (This should be the similar to the project plan)

Figure 1 - Task Planning Diagram

Figure 2 - Gantt chart

Figure 3 - Risk Management Table

Figure 4 - Societal Needs Table

Figure 5 - Weighted Decision Matrix

1 Team

1.1 TEAM MEMBERS

Bryce Carter, James Gibbs, Theodore Tayib, Chee Hau Fong, Kyoungken Lee, Derrick Brandt

1.2 REQUIRED SKILL SETS FOR YOUR PROJECT

This project requires skills with full stack development, DevOps, cloud, and hardware understanding.

1.3 SKILL SETS COVERED BY THE TEAM

The skills required for this project are meant to be learned because it is an exploratory design endeavor. Nevertheless, all members have full stack experience. Derrick Brandt has cloud and DevOps experience. Theodore Thayib and James Gibbs have hardware understanding.

1.4 PROJECT MANAGEMENT STYLE ADOPTED BY THE TEAM

We have adopted an agile project management style.

1.5 INITIAL PROJECT MANAGEMENT ROLES

Bryce Carter	Individual Component Design
Derrick Brandt	Team Organization and Communications
Kyoungeun Lee	Goal Setting and Motivation
Chee Hau Fong	Testing
James Gibbs	Team Lead
Theodore Thayib	Client Interaction

2 Introduction

2.1 PROBLEM STATEMENT

Docker, Kubernetes, and cloud platforms are causing a paradigm shift in the software engineering profession. It would benefit students to have the ability to investigate the capabilities of this new era. An exploratory senior design project could allow students to confront their information gap through research. Additionally, it could allow them to document their findings in building a miniature cloud platform with Turing Pi, Docker, and Kubernetes.

2.2 INTENDED USERS AND USES

Intended users are students, including Iowa State University, who are interested in affordable power-efficient cluster machines and beginners for edge computing/kubernetes/Docker.

2.3 REQUIREMENTS & CONSTRAINTS

Functional requirements:

1. The senior design product shall implement a scalable message service.
2. While the Turing Pi is online, the senior design product shall use an open source distributed event streaming platform.
3. The senior design project shall have three compute modules available at any given time.

Non-functional requirements:

1. The service landing page should maintain smooth performance scalable to 30,000 requests per second.
2. The messaging service shall propagate messages to user groups within .75 seconds.
3. If the messaging service is under 90% load or greater, then the service shall propagate messages to user groups within 2 seconds.
4. The service shall be available to United States users 98.99% of the month.
5. The messaging service shall encrypt user messages in transit.

Resource requirements:

1. The senior design product shall use the Hypriot operating system.
2. The senior design product shall use Docker as its containerization platform.
3. The senior design product shall use Kubernetes as its orchestration platform.
4. The senior design product shall use CI/CD pipelines for building and testing code.

Physical Requirements:

1. The turing pi product shall be placed inside a mini ITX case.
2. The senior design product shall use a minimum of three raspberry pi compute modules.

UI Requirements:

1. The senior design product shall implement a simple graphical user interface.

2.4 ENGINEERING STANDARDS

The senior design product shall use GraphQL messaging protocol.

The users will benefit from performance increase, a key factor in user enjoyment of messaging services.

The messaging service shall use HTTPS encryption protocols.

Security of the application is important enough to warrant a modern standard. Failure to use HTTPS is simply unacceptable in the modern era.

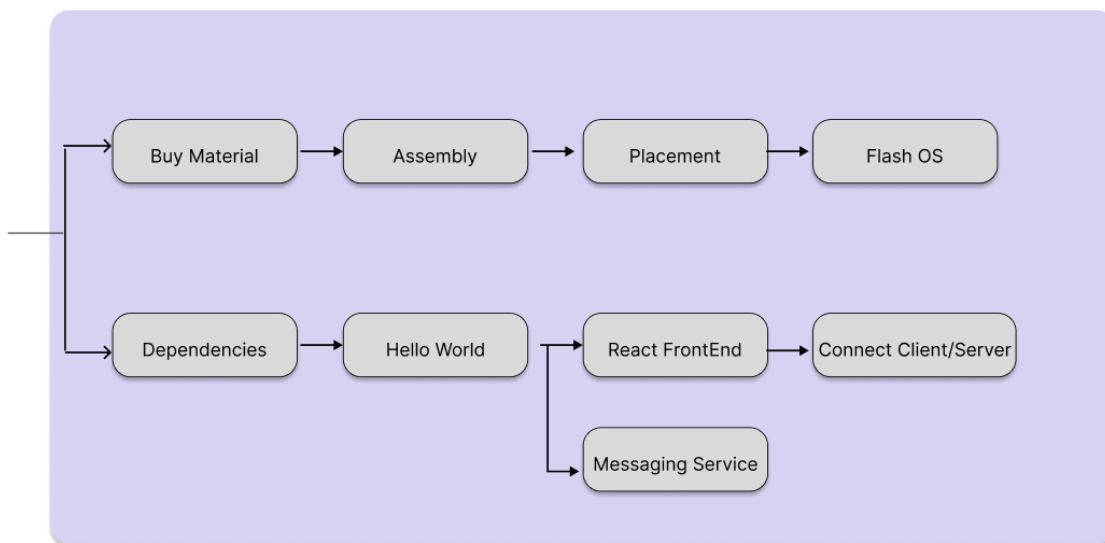
3 Project Plan

3.1 PROJECT MANAGEMENT/TRACKING PROCEDURES

This project shall use an agile approach. User stories and short sprints will provide us the flexibility needed to properly implement the requirements. Additionally, We will use Git for version control and GitLab for remote repository management and work tracking.

3.2 TASK DECOMPOSITION

We have decomposed the major tasks into 9 steps and turned them into a diagram according to the project's timeline.



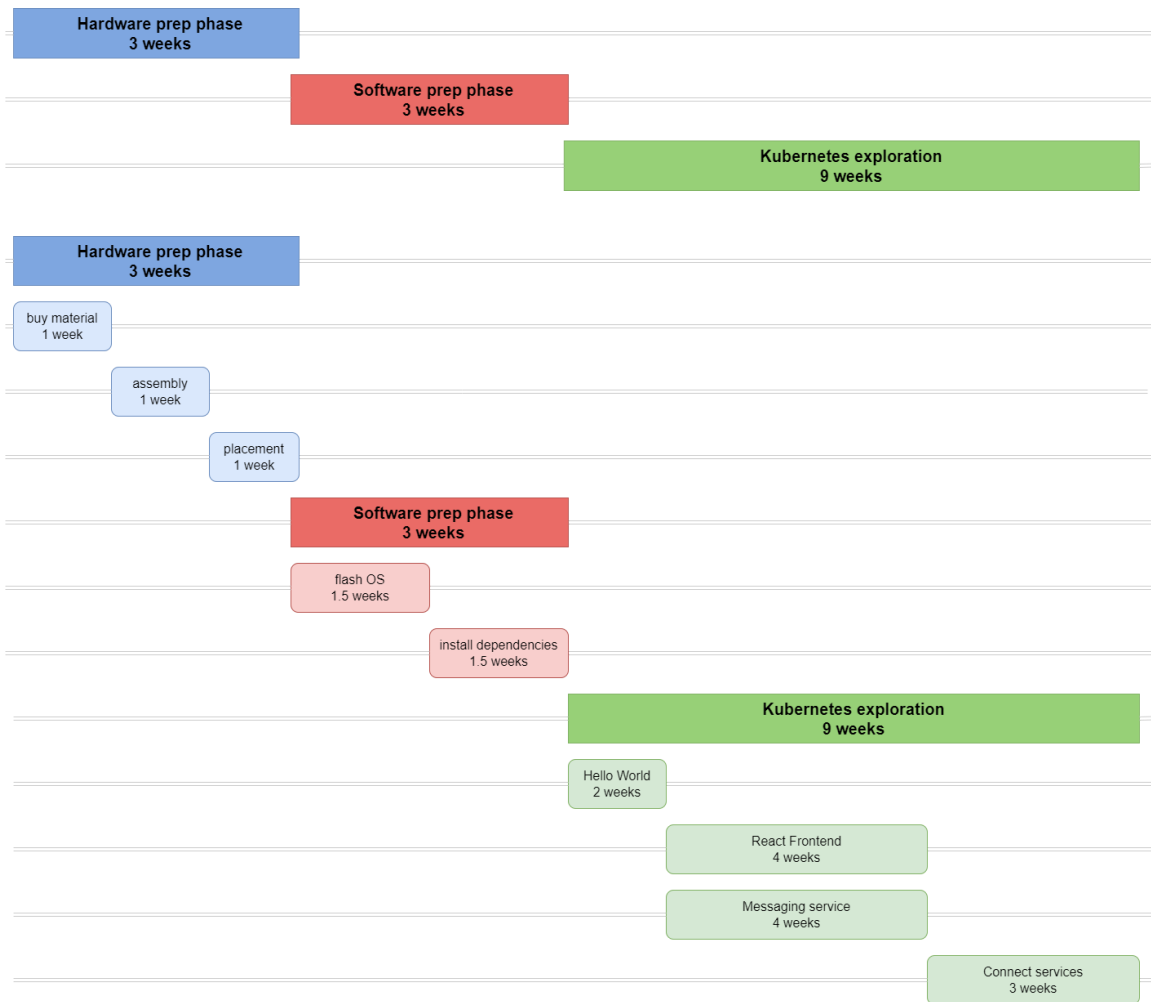
3.3 PROJECT PROPOSED MILESTONES, METRICS, AND EVALUATION CRITERIA

The project has three major milestones: Flashing the Hyprion OS onto the raspberry pi compute modules; creating the simple frontend for the messaging service; connecting the front end and the backend.



3.4 PROJECT TIMELINE/SCHEDULE

The project has three broad phases. These phases are further broken down below.



3.5 RISKS AND RISK MANAGEMENT/MITIGATION

Please find the risks and their mitigation in the table below.

Risks		Probability	Consequence	Mitigation Plan
Supplier risk	Out of Stock	High	Schedule Delays	Order early
	Delivery Delay			

Health & safety	Personal Injury	Low	Harmed Students	Don't rush the hardware assembly
Project Complexity	Information Silos; Asymptotic Performance	Moderate	Productivity Impact; Scope Reduction	Focus on simple design; document document document
Schedule risk	Whiffed estimates	High	Schedule Changes	Agile methodology

3.6 PERSONNEL EFFORT REQUIREMENTS

Task 1: Three persons: 5 hours

Task 2: Three persons: 5 hours

Task 3: Three persons: 5 hours

Task 4: Three persons: 7 hours

Task 5: Three persons: 7 hours

Task 6: All team: 10 hours

Task 7/8: Two teams of three each: 20 hours

Task 9: Two teams of three each: 20 hours

3.7 OTHER RESOURCE REQUIREMENTS

N/A

4 Design

4.1 DESIGN CONTEXT

4.1.1 Broader Context

We are designing mainly for ourselves, as this is an exploratory project meant for a senior design team to learn about these important technologies. We have also spent time considering a broader context where our project might place itself.

Area	Description	Examples
Public health, safety, and welfare	How does your project affect the general well-being of various stakeholder groups? These groups may be direct users or may be indirectly affected (e.g.,	Our project does not affect the health or well-being of the public directly. Indirect effects would be the result of surprises born from the technology.

	solution is implemented in their communities)	
Global, cultural, and social	How well does your project reflect the values, practices, and aims of the cultural groups it affects? Groups may include but are not limited to specific communities, nations, professions, workplaces, and ethnic cultures.	Exploration into burgeoning technology is braided with the university experience for many students. This project is a good reflection of a research university's values.
Environmental	What environmental impact might your project have? This can include indirect effects, such as deforestation or unsustainable practices related to materials manufacture or procurement.	This project will have minimal environmental impact attributable to energy used to power electronics.
Economic	What economic impact might your project have? This can include the financial viability of your product within your team or company, cost to consumers, or broader economic effects on communities, markets, nations, and other groups.	This project will have minimal economic impact on Iowa State University. The project has a low likelihood of having an outsized impact economically pending exceptional outcomes.

4.1.2 Prior Work/Solutions

Docker and Kubernetes are the premier products on the market. These technologies have been used in the industry for some time and documentation on them is extensive. Docker containerization is utilized by developers for many projects and works great with other applications like Jenkins, Jfrog, and Gitlab. Using edge computing, containerization, and cloud orchestration allows for better asynchronous functionality and distributed computing. Which is perfect for implementing a messaging service that updates information continuously and has many devices requesting as well as sending information. This project is not looking to create some entirely new solution but to see what can be learned from existing products.

<https://kubernetes.io/>

<https://www.docker.com/>

<https://turingpi.com/>

4.1.3 Technical Complexity

This project requires understanding of difficult software products and their interaction. Multiple open source software need to interact with our own proprietary system to deliver the end product promised. The open source software in question, Docker, Kubernetes, and Kafka, are of sufficient newness that no current solution allows students to learn these in a classroom environment. Therefore, building a system with complex components where no education currently exists is of sufficient difficulty to meet any technically demanding requirement.

4.2 DESIGN EXPLORATION

4.2.1 Design Decisions

1. Use of Turing Pi hardware

The usage of the Turing Pi platform is essential to meeting the client's vision.

2. Use of Kubernetes for orchestration

Kubernetes is an industry standard container orchestration platform. This exploration of the platform will enable future students to be better prepared for current and future job opportunities.

3. Use of Apache Kafka for event streaming

Kafka will best fit the project because of its unique queuing approach to disparate sourced information.

4.2.2 Ideation

We ideated using a technique titled "lotus blossom." This led us to new potential pathways. We considered the use of RabbitMQ as an alternative to Apache Kafka. We also considered what might happen if we used Docker Swarm instead of Kubernetes. Besides the Turing Pi, we had to consider a barebones motherboard-less option where we sync up everything ourselves. Unrelated to our main design decision, we talked about what version control system we should use, and how we were going to use CI/CD in this project.

4.2.3 Decision-Making and Trade-Off

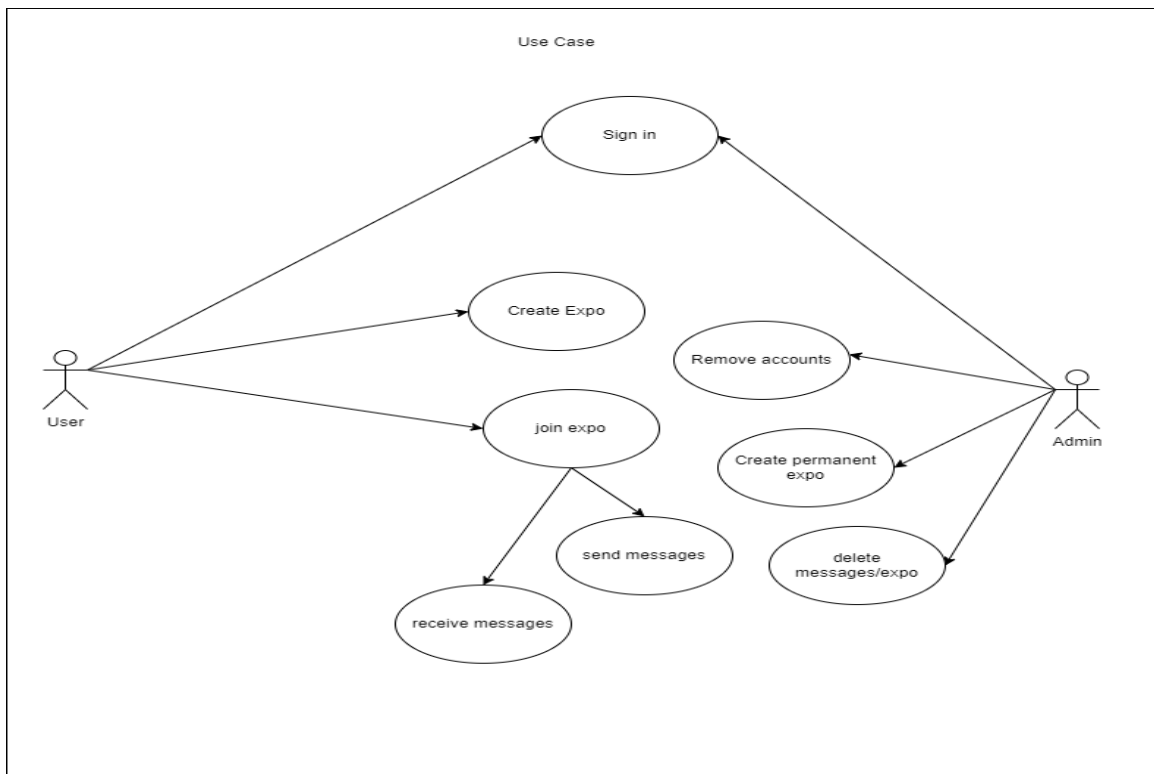
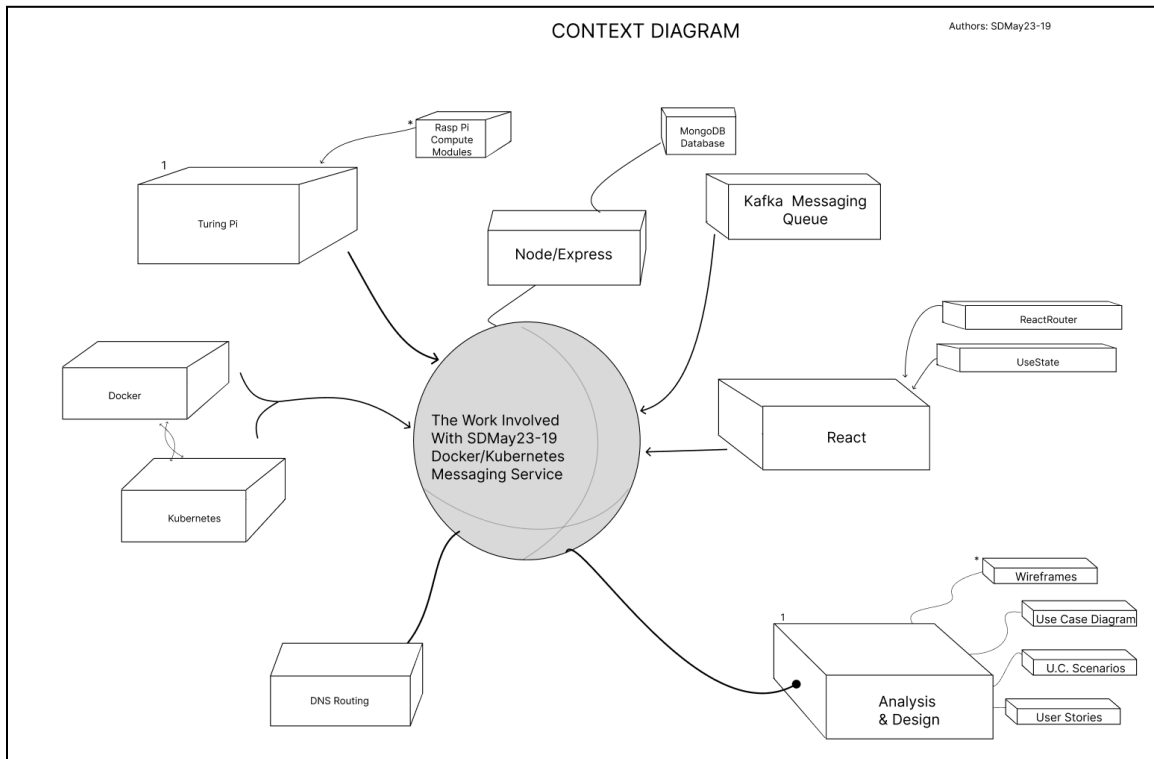
Selection Criteria	Criterion Weight	Messaging Service	AI Image Recognition	Ecommerce
Complexity	.2	4	8	4
Kubernetes Applicability	.3	8	2	5
Previous Experience	.5	8	2	3

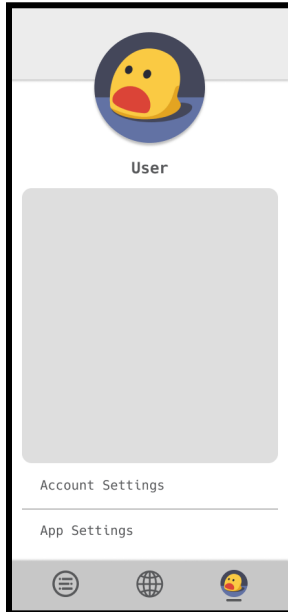
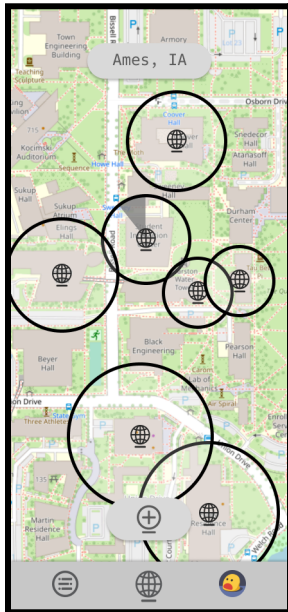
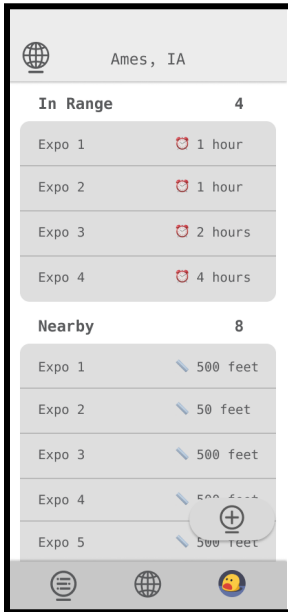
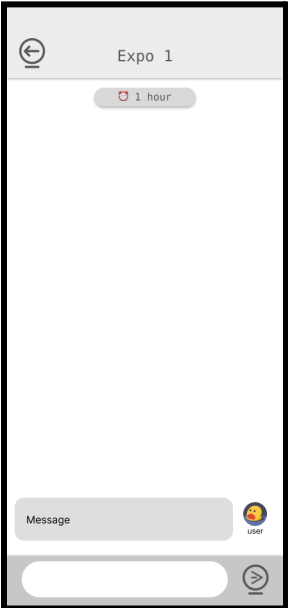
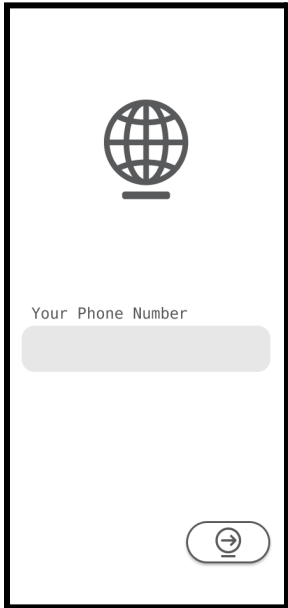
4.3 PROPOSED DESIGN

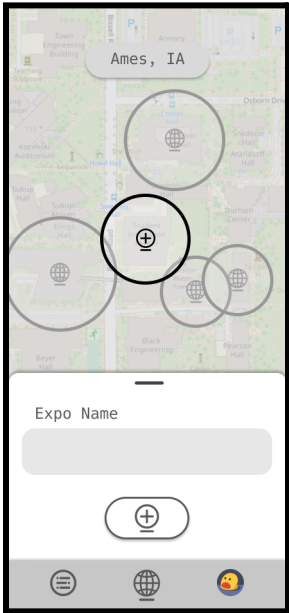
4.3.1 Overview

The project involves an online messaging service where users can enter chat rooms and communicate with one another on campus. The current design involves creating more than one chat channel that is region locked.

4.3.2 Detailed Design and Visual(s)







Web Application Side:



4.3.3 Functionality

Users who are on Iowa State campus would be capable of accessing our application. From there, they could enter chat rooms based on their location on campus. Once inside the chat room, the users may talk with anyone else also in that region's space. Any messages sent into the chat are propagated to others in that chat room.

4.3.4 Areas of Concern and Development

The design of the application satisfies the requirements and meets the user needs fairly. The application's design is complex enough to utilize the technologies the project is meant to investigate which is the goal of the design. The main concern for the system's current design will be how scalable it can be. Scalability is a primary functionality of the services/technologies in the system.

4.4 TECHNOLOGY CONSIDERATIONS

Since this is an exploratory project for Turing Pi, Docker, and Kubernetes, there are only a few considerations to be made. For hardware, we have decided to use Turing Pi 2 since Turing Pi has been discontinued, and it also allows us to have more options and better and newer hardware for our compute modules. For the compute modules, we mainly considered the price, memory, storage, and connectivity. Ultimately, we chose CM4104016, which has 4GB RAM, 16GB of eMMC storage, and wireless connectivity like Wi-Fi and Bluetooth.

For our event streaming platform, we have considered two different platforms. They are RabbitMQ and Apache Kafka. RabbitMQ is one of the most commonly used message brokers, allowing applications and services to communicate data without requiring uniform exchange protocols. On the other hand, Kafka is generally used to create real-time streaming data pipelines and adaptive applications. It combines communications, storage, and stream processing to enable storing and analyzing historical and real-time data. Both event streaming platforms are open source and can process millions of messages simultaneously. However, we decided to use Kafka as it is more efficient than RabbitMQ with the same hardware.

4.5 DESIGN ANALYSIS

We have gotten a development environment set up inside of docker. Additionally, we have gotten some preliminary wireframes set up, as seen above. Not seen above is an itemized list with hardware we need for the project.

5 Testing

Testing is an **extremely** important component of projects, whether it involves a circuit, a process, power system, or software.

5.1 UNIT TESTING

The project shall use JEST. JEST is a simple Javascript testing framework. It's fast and safe, reliably running multiple tests in parallel. It also allows for mocking, as well as easy to comprehend exceptions when tests fail. We'll be able to use it with both our react frontend, and express backend.

5.2 INTERFACE TESTING

React can interface with JEST described above. This allows us to test the front end interfaces. Our backend can be tested with JEST because it is javascript based.

5.3 INTEGRATION TESTING

Our critical paths involve backend containers orchestrated by kubernetes. We plan on testing this with K6, a technology that measures Kubernetes metrics, a key technology required in the project. Contract testing will be used to test individual services and then integration and component testing will be used to test kubernetes orchestration and kafka channels.

5.4 SYSTEM TESTING

K6 will be used to load test the kubernetes deployment,

5.5 REGRESSION TESTING

New additions are ensured not to break through CI/CD pipeline integration.

5.6 ACCEPTANCE TESTING

User acceptance testing phase after sprints.

5.7 SECURITY TESTING (IF APPLICABLE)

N/A

5.8 RESULTS

The tests are designed to assist in our goal of exploring the functionality of the technologies being used. While it is important to ensure behavior of the end product, our capstone project focuses on diving into technologies Kubernetes and Docker. The focus is on education, so testing will help us toward that end.

6 Implementation

Spring semester will bring the building of the Turing Pi. Additionally, next semester we will implement all container and orchestration systems. Web development and application software and begins the second semester as well. With the container and orchestration implemented and linked with the messaging service, the cloud platform hosted on the Turing Pi can be complete.

7 Professional Responsibility

7.1 AREAS OF RESPONSIBILITY

Area of Responsibility		
Work Competence	Perform work of high quality, integrity, timeliness, and professional competence.	Perform services only in areas of their competence; Avoid deceptive acts.
Financial Responsibility	Deliver products and services of realizable value and at reasonable costs.	Act for each employer or client as faithful agents or trustees.
Communication Honesty	Reports work truthfully, without deception, and are understandable to stakeholders.	Issue public statements only in an objective and truthful manner; Avoid deceptive acts.
Health, Safety, Well-Being	Minimize risks to safety, health, and well-being of stakeholders.	Hold paramount the safety, health, and welfare of the public.
Property Ownership	Respect property, ideas, and information of clients and others.	Act for each employer or client as faithful agents or trustees.
Sustainability	Protect the environment and natural resources locally and globally.	Balance human needs and the environment.
Social Responsibility	Produce products and services that benefit society and communities.	Conduct themselves honorably, responsibly, ethically, and lawfully to enhance the honor, reputation, and usefulness of the profession.

7.2 PROJECT SPECIFIC PROFESSIONAL RESPONSIBILITY AREAS

Area of Responsibility	Rating	Reasoning
Work Competence	High	The research conducted by this project will be used to influence teaching of future and emerging technology at Iowa State and thus must be of the highest quality.
Financial Responsibility	Medium	This project must also balance the financial needs of the university.
Communication Honesty	High	Documentation is essential to inform stakeholders of research conclusions.
Health, Safety, Well-Being	Low	Exploratory programs must put the well-being of users first.
Property Ownership	High	Ownership of hardware used by the university must be respected.
Social Responsibility	Medium	Exploratory programs must respect the public good.

7.3 MOST APPLICABLE PROFESSIONAL RESPONSIBILITY AREA

The most applicable area of responsibility is work competence. The members of the team must all work together to bring this exploration to life. The scale is too large, so it is important that work competence responsibilities are honored and kept.

8 Closing Material

8.1 DISCUSSION

The main focus of this project is exploring the technologies changing the face of software engineering. Docker, Kubernetes, and the cloud alter what it means to run “at scale.” The results and lessons learned will come from our experimentation next semester when we implement our design.

8.2 CONCLUSION

At this stage in development, we have a Docker development environment set up and interfaced with Kafka and Kubernetes. Supply chain issues and chip shortages bar us from starting work on the Turing Pi.

8.3 REFERENCES

S. Cass, "Home Clustering Made Easier: Learn Docker and Kubernetes with the Turing Pi," in *IEEE Spectrum*, vol. 58, no. 11, pp. 16-18, November 2021, doi: 10.1109/MSPEC.2021.9606504. Available at: <https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=9606504>

8.4.1 Team Contract

Team Members:

- | | |
|-------------------------------|--------------------------------|
| 1) _____ Bryce Carter _____ | 2) _____ Chee Hau Fong _____ |
| 3) _____ Derrick Brandt _____ | 4) _____ James Gibbs _____ |
| 5) _____ Kyoungkeun Lee _____ | 6) _____ Theodore Thayib _____ |

Team Procedures

1. Day, time, and location (face-to-face or virtual) for regular team meetings:

The Team shall hold meetings weekly on Tuesday from 5:30 pm to 6:30 pm over Discord. *Subject to change by supermajority consensus.*

2. Preferred method of communication updates, reminders, issues, and scheduling (e.g., e-mail, phone, app, face-to-face):

The Team shall prefer Discord, the instant messaging social platform, as the communication method. *Subject to change by supermajority consensus.*

3. Decision-making policy (e.g., consensus, majority vote):

Decision-making procedure:

- (a) Consensus vote for small choices for members present
- (b) Polls for large decisions with team lead as tie-breaker

4. Procedures for record keeping (i.e., who will keep meeting minutes, how will minutes be shared/archived):

Record keeping procedure:

- (a) Meeting Agenda Kept in Google Drive
- (b) Meeting Minutes Kept in Google Drive
- (c) Meeting Agenda Posted in Discord Channel #meeting-records
- (d) Meeting Minutes Posted in Discord Channel #meeting-records

- (a) Decisions made for small choices are entered into a Team Decisions document with a justification statement. Example: We chose this route for **x** reason.
- (b) Large decisions are entered into the Team Decisions documents as a screenshot of the poll.

Participation Expectations

1. Expected individual attendance, punctuality, and participation at all team meetings:

Expectations shall include (1) **on time** and (2) **prepared** unless extenuating circumstances prevent the member.

2. Expected level of responsibility for fulfilling team assignments, timelines, and deadlines:

Expectations shall include (1) **equal effort** and (2) **completion of work** unless extenuating circumstances prevent the member.

3. Expected level of communication with other team members:

Expectations shall include (1) **24 hour reply time** and (2) **response in discord demonstrating up-to-date knowledge** unless extenuating circumstances prevent the member.

4. Expected level of commitment to team decisions and tasks:

Expectations shall include (1) **equal consideration of ideas** and (2) **trust in team decisions** unless extenuating circumstances prevent the member.

Leadership

1. Leadership roles for each team member (e.g., team organization, client interaction, individual component design, testing, etc.):

Bryce Carter: Individual Component Design

Derrick Brandt: Team Organization and Communications

Kyoungkeun Lee: Goal Setting and Motivation

Chee Hau Fong: Testing

James Gibbs: Team Lead

Theodore Thayib: Client Interaction (*To Be Confirmed*)

2. Strategies for supporting and guiding the work of all team members:

Standard Operating Strategy:

- (a) Analyze the problems inherent with the work
- (b) Estimate the time commitment of the work
- (c) Delegate the work

3. Strategies for recognizing the contributions of all team members:

Standard Operating Strategy:

- (a) Start meetings with shoutouts for great work

Collaboration and Inclusion

1. Describe the skills, expertise, and unique perspectives each team member brings to the team.

Bryce Carter: Experience with development operations, and working with different teams in an industry environment.

Derrick Brandt: Experience with cloud platforms, serverless functions, APIs, CI/CD pipelines, automated cloud architecture provision.

Kyoungkeun Lee: Experience with embedded devices and basic knowledge of operating systems and I/O systems.

Chee Hau Fong: Experience with APIs and CI/CD

James Gibbs: Experience with docker and kubernetes

Theodore Thayib: Experience with docker, APIs, javascript (react), backend (flask)

2. Strategies for encouraging and supporting contributions and ideas from all team members:

Standard Operating Strategy:

- (a) Encourage others to give input if they haven't had a chance to talk much.

3. Procedures for identifying and resolving collaboration or inclusion issues (e.g., how will a team member inform the team that the team environment is obstructing their opportunity or ability to contribute?)

Standard Operating Strategy:

- (a) Member in command has responsibility for maintaining respectable group dynamics

Goal-Setting, Planning, and Execution

1. Team goals for this semester:

The Team goals include:

- (a) Completing all assignments and turn in on time
- (b) Finishing project design and implementation
- (c) Creating teamwork

2. Strategies for planning and assigning individual and team work:

Standard Operating Strategy:

- (a) Discussing as a team about assigning work based on member's individual skills
- (b) Rewarding how much they achieve according to weekly goal
- (c) Sharing progress with team members weekly

Exceptional Conditions Strategy:

- (a) Give an optional week break every 4 weeks
- (b) Impromptu team meetings
- (c) Assigning specific tasks

3. Strategies for keeping on task:

Standard Operating Strategy:

- (a) Progress reports on weekly meeting
- (b) Asking help if stuck and being available to team
- (c) Meeting minutes for keeping track and feedback

Exceptional Conditions Strategy:

- (a) Stricter Agile/Scrum documentation
- (b) More consistent reminders
- (c) More consistent progress reports

Consequences for Not Adhering to Team Contract

1. How will you handle infractions of any of the obligations of this team contract?

Misdemeanor Procedure:

- (a) Ask if everything in their life is going okay
- (b) Ask if there is anything we can do to help
- (c) Reduce complexity, within reason, for next task

2. What will your team do if the infractions continue?

Major Transgression Procedure:

- (a) Notify Professor
- (b) Notify student that professor has been notified

a) I participated in formulating the standards, roles, and procedures as stated in this contract.

b) I understand that I am obligated to abide by these terms and conditions.

c) I understand that if I do not abide by these terms and conditions, I will suffer the consequences as stated in this contract.

1) Bryce Cater DATE 9/23/2022

2) James Gibbs DATE 9/23/2022

3) Chee Hau Fong DATE 9/23/2022

4) Derrick Brandt DATE 9/23/2022

5) Kyoungkeun Lee DATE 9/23/2022

6) Theodore Thayib DATE 9/23/2022