Knockk

Grace Radlund

CST-451 Capstone Project Proposal

Grand Canyon University

Instructor: Professor Mark Reha

Revision: 1.4

Date: 4/24/2024

**ABSTRACT**

Although social media is prominent, there is no social network to connect residents of an apartment building. This project aims to fill that void with Knockk. Knockk is a social network composed of two applications: a mobile application and a web application. The mobile application is designed for residents, with the primary goal of connecting the resident with neighboring units. The web application is for building administrators to verify and manage residents of a building. Although this project can be defined as a social network, it is not like most other social applications – social media postings are omitted. This project solely focuses on connecting residents together and is not focused on being a comparison game of likes, comments, and followers.

To make this project successful, the project is laid out in terms of planning, analyzing, designing, and developing, with a completion date of May 3, 2025. This document is the proposal for this project and is in the planning phase. A vague description in terms of what the project consists of has been developed, with more technical terminology in the high-level solution. Very technical terminology has been omitted. Basic functionality of the applications will be completed before any other features are pulled in to ensure the project can be completed with the success criteria met. Since this project is a large undertaking, there is an in-depth risk management plan to minimize issues occurring in this project. Proposed technologies will be researched, and proof of concepts will be developed before a final decision on the technology stack is made.

|  |
| --- |
| History and Signoff Sheet |

**Change Record**

|  |  |  |
| --- | --- | --- |
| **Date** | **Author** | **Revision Notes** |
| 9/22/2024 | Grace Radlund | Initial draft for review/discussion |
| 10/22/2024 | Grace Radlund | Updated missing Requirements Phase in WBD |
| 4/8/2025 | Grace Radlund | Preparing document for showcase |
| 4/24/2025 | Grace Radlund | Preparing for project portfolio |

|  |
| --- |
| **Overall Instructor Feedback/Comments** |

|  |
| --- |
| **Overall Instructor Feedback/Comments** |

**Integrated Instructor Feedback into Project Documentation**

Yes  No

**Project Approval**

Professor Mark Reha

**TABLE OF CONTENTS**

Project Overview and Project Objectives 5

Project Scope 9

Project Success Measures 12

Project High-Level Solution 14

Project Controls 17

Project Cost and Schedule 19

Appendix A – References 20

Appendix B – Copyright Compliance 21

Project Overview and Project Objectives

**State the Problem and Background**

In a survey conducted in 2021, “Nearly 1 in 6 people don’t know any of their neighbors’ names” (Gee, 2021). A problem currently exists where people do not know whom they live next to. Why is that? One reason is that there currently is not a social network for residents in building complexes. This project aims to change that. The purpose of this project, titled Knockk, is to create a social network that connects residents living in the same apartment building together. The goal is to design two applications: a mobile application for residents (synonymous in this proposal with the term users) and a web application for building administrators. The main objective of the mobile application is to let residents’ view who lives around them. The web application allows building administrators to manage and verify residents. Verification is important so that residents cannot claim they live in a unit that they do not. It isn’t until the resident is verified, that they can login to the app.

While this is a social application, Knockk ensures proper privacy for residents. Once a resident registers, they will only be able to see the first names of those in a unit they neighbor with. Only until they connect with a fellow resident will they see more information. This connection is established when a resident sends a connection request, which the recipient resident then accepts. Information given by the user upon account creation will then be visible to the other resident. This information will be personal data such as their full name, age, gender, biography, social media usernames, etc. In lieu of privacy, residents do not have to enter any personal information other than their full name.

**Christian Worldview**

Mark 12:30-31 says, “Love the Lord your God with all your heart and with all your soul and with all your mind and with all your strength.’31 The second is this: ‘Love your neighbor as yourself.’ There is no commandment greater than these” (*New International Version*, 1973/2011, Mark 12:30-31). The bible commands us to love your neighbor as yourself, but most people do not know their neighbors. With Knockk, residents can get to know who lives around them and connect to form Godly relationships that will benefit God’s kingdom.

**Project Objectives**

The project will research, plan, and develop a social application using a comprehensive tech stack to create a network for residents of apartment buildings. To ensure technologies fit the project, research and proof of concepts will be developed.

Two user facing applications and an API will be developed: a mobile application for residents, a web application for building administrators, and an API that both user facing applications will consume. The API will communicate to a database that will store data such as resident information and friendships.   
If objectives are met, enhancements to the project will be pursued which include: a chat feature for residents to communicate within the app, viewing all residents of the apartment building (in addition to the units they neighbor with), push notifications for building alerts (like fire alarms) and deploying the mobile application to the App and Google Play stores.

**Challenges**

Nothing worthwhile is easy, and Knockk is no exception. There are challenges to planning, designing, and developing Knockk, which include:

* Team size.

Knockk is currently a one-person team. This creates the possibility of tunnel vision on decisions and slower sprint velocities compared to what a larger team could achieve. Multiple mentors will be sought out to provide valuable insight and advice to combat this challenge.

* Selecting a tech stack.

There are many technologies to choose from, which can get overwhelming. Once the technology stack is finalized, the plan should be followed unless a compelling reason arises.

* Beta testing and attracting users.

For the network to perform as expected, the majority of residents will have to be on the network. If a resident does not register, they will be unknown to other residents who are registered.

* Proper architecture.

If the network is successful, it must have proper architecture so that it can scale.

* Following best practices.

There are many “best” practices and standards – seeking industry standards will be pursued. Lessons learned from Liberty Mutual and Grand Canyon University will be brought in by the team.

* Proper data management.

Designing a database to store data accurately and efficiently.

* Applications must function effectively.

For the resident application to work properly, the administrative application must be functioning as residents need to be verified by the admin before they can sign-in to the resident application. This means both applications must be fully functioning.

These challenges will be overcome.

**Benefits and Opportunities**

Knockk provides residents in an apartment building the opportunity to connect with their neighbors. Too often social networks revolve around the idea of likes or followers. While Knockk is considered a social network since residents are connecting with one another, it is not like most social applications. There are no user posts of any type (including likes and comments), which are primary elements that drive social comparisons in social applications. In today’s society, a person’s likeability is often tied to the presence of their social media following, which is a false sense of true likeability, especially because followers can be purchased. Knockk aims to be different; not a typical social network, but simply a way to connect with their neighbors.

Project Scope

Knockk is a social network that solves the problem residents have of not knowing who they live next to. Below are features and capabilities that will help solve the problem. In scope are features and capabilities that will be completed for the first phase of this project. Out of scope are features and capabilities the team wants to develop if everything in scope is completed and/or during the next phase. Below in and out of scope is the work breakdown structure for this project with tasks that satisfy the project’s objectives.

**In Scope**

* Creation of accounts by residents
* Verification of residents by building admin
* Managing residents by building admin
* Editing of accounts by residents
* Connecting with other residents
* Displaying who lives above a resident
* Displaying who lives to the right of a resident
* Displaying who lives below a resident
* Displaying who lives to the left of a resident

**Out of Scope**

* Chatting with other residents
* Push notifications for building alerts by building admin
* Posting of building news and events by building admin
* Displaying building news and events
* Searching for other residents in the building
* Making the API public facing
* Securing the API
* Hosting the admin website
* Expanding to home communities (and not just apartment buildings)
* Publishing the mobile app to the App and Google Play stores

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Work Breakdown Structure | | | | | | | | | | |
| ID | Task | Dependencies | Status | Effort Hours | Cost | Start Date | Planned Completion | Estimate to Completion | Actual Completion | Resource |
| 0 | Project Planning | Initial idea | Complete | 25 | 150 CST-451 points | 9/10/2024 | 9/22/2024 | 25 | 25 | Grace |
| 1 | React Native proof of concept | Initial idea | In-progress | 20 |  | 9/10/2024 | 10/14/2024 | 20 | 20 | Grace, React Native documentation |
| 2 | React proof of concept | Initial idea | In-progress | 20 |  | 9/10/2024 | 10/14/2024 | 20 | 20 | Grace, React documentation |
| 3 | Flutter proof of concept | Initial idea | In-progress | 20 |  | 9/10/2024 | 10/14/2024 | 20 | 20 | Grace, Flutter documentation |
| 4 | Express proof of concept | Initial idea | In-progress | 20 |  | 9/10/2024 | 10/14/2024 | 20 | 20 | Grace, Express documentation |
| 5 | Project Analysis/Requirements | Planning | Planned | 40 | 150 CST-451 points | 9/23/2024 | 10/13/2024 | 40 | 45 | Grace |
| 6 | Project Design | Planning, Analysis, and MVP | Planned | 40 | 450 CST-451 points | 10/14/2024 | 11/17/2024 | 40 | 60 | Grace |
| 7 | Code Development, Testing, and Implementation | Planning, Analysis, and Design Complete | Planned | 40 | 100 CST-451 points | 11/18/2024 | 12/15/2024 | 40 | 40 | Grace |
| 8 | Code Development, Testing, and Implementation | Planning, Analysis, Design Complete, and Previous Development | Planned | 40 | 150 CST-452 points | 1/6/2025 | 2/15/2025 | 40 | 40 | Grace |
| 8 | Code Development, Testing, and Implementation | Planning, Analysis, Design Complete, and Previous Development | Planned | 40 | 150 CST-452 points | 2/16/2025 | 3/29/2025 | 40 | 40 | Grace |
| 10 | Code Development, Testing, and Implementation | Planning, Analysis, Design Complete, and Previous Development | Planned | 40 | 150 CST-452 points | 3/30/2025 | 5/3/2025 | 40 |  | Grace |
| 11 | Benchmark Final Project | Planning, Analysis, Design Complete, and Development | Planned | 40 | 450 CST-452 points | 3/30/2025 | 5/3/2025 | 40 |  | Grace |
| 12 | Final Project Presentation | Planning, Analysis, Design Complete, and Development | Planned | 10 | 100 CST-452 points | 4/28/2025 | 5/3/2025 | 10 |  | Grace |
| 13 | Research and develop a proof of concept with JWTs | Project in-scope features complete | Planned | 20 |  | Once in-scope features are complete | 5/3/2025 | 20 |  | Grace |
| 14 | Research and develop a proof of concept with real time databases | Project in-scope features complete | Planned | 20 |  | Once in-scope features are complete | 5/3/2025 | 20 |  | Grace |
| 15 | Research how to make an API public facing | Project in-scope features complete | Planned | 20 |  | Once in-scope features are complete | 5/3/2025 | 20 |  | Grace |
| 16 | Development of out-of-scope features | Project in-scope features complete | Planned | 100 |  | Once in-scope features are complete | 5/3/2025 | 100 |  | Grace |

Project Success Measures

The goal of this project is to have every feature listed in scope completed on time. The table below shows what will be used to calculate the project’s success based on its main goals. Below the completion criteria is a list of assumptions and constraints for this project. The exact technology stack at the time is still unknown, so vague high-level descriptions have been made. Proposed technologies can be found in the project high-level solution with the risks of choosing each listed in the risk management plan. Technologies unknown by the team will be researched and a proof of concept will be developed before the final decision is made. If a technology does not fit the project, a new one will be sought out.

|  |
| --- |
| Project Completion Criteria |
| 1 – A resident can sign up for an account |
| 2 – An admin user can verify and activate resident accounts |
| 3 – A resident can see whom lives above, below, and to the sides of them |
| 4 – A resident can connect with a fellow neighbor |
| 5 – A resident can edit their profile |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Assumptions and Constraints | | | | | |
| ID | Description | Comments | Type | Status | Date Entered |
| 1 | Assuming a mobile application will be built for residents. | See risk management plan | Assumption | OK | 9/20/24 |
| 2 | Assuming a web admin application will be built for building administration. | See risk management plan. | Assumption | OK | 9/20/24 |
| 3 | Assuming an API will be built. | See risk management plan. | Assumption | OK | 9/20/24 |
| 4 | Assuming a database will store necessary data. | See risk management plan. | Assumption | OK | 9/20/24 |
| 5 | Assuming GitHub will be used as a version control software for free. | See risk management plan. | Assumption | OK | 9/20/24 |
| 6 | Assuming Jira will be used for project management for free. | See risk management plan. | Assumption | OK | 9/20/24 |
| 8 | Assuming Draw.io will be used for technical diagramming for free. | See risk management plan. | Assumption | OK | 9/20/24 |
| 8 | An Apple Developer id and Google Developer account will have to be bought to deploy this application. | TestFlight needs a developer id for testing.  See risk management plan. | Constraint | OK | 9/20/24 |
| 9 | Azure allows for 12 months free or $100 credit, whichever expires first. | Initially out of scope.  See risk management plan. | Constraint | OK | 9/20/24 |
| 10 | Assuming Figma’s student account will provide free Figma for the project’s duration. | See risk management plan. | Assumption | OK | 9/20/24 |
| 11 | Assuming JWTs will secure the API. | Initially out of scope.  See risk management plan. | Assumption | OK | 9/20/24 |
| 12 | Assuming a real time database will provide chatting between users. | Initially out of scope.  See risk management plan. | Assumption | OK | 9/20/24 |
| 13 | Assuming push notifications will notify users of building alerts. | Initially out of scope.  See risk management plan. | Assumption | OK | 9/20/24 |
| 14 | Assuming building administrators can post building news and events. | Initially out of scope.  See risk management plan. | Assumption | OK | 9/20/24 |
| 15 | Assuming the API will be public facing. | Initially out of scope.  See risk management plan. | Assumption | OK | 9/20/24 |
| 16 | Assuming the administration app will be hosted. | Initially out of scope.  See risk management plan. | Assumption | OK | 9/20/24 |
| 17 | Assuming the web application can be hosted. | Initially out of scope. | Assumption | OK | 9/20/24 |
| 18 | Cloud Provider Failure | See risk management plan. | Constraint | OK | 9/20/24 |

Project High-Level Solution

**Introduction**

Knockk is a social networking project that aims to connect neighbors with one another. The main objective is to create a mobile application for residents to get to know their neighbors. Residents will be able to connect with one another through connection requests, and once accepted, will be able to view more information on the resident. There will also be a web application for building administrators to verify and manage residents. The two applications will consume an API that will communicate with a relational database which will store data.

**Assumptions**

To complete the project, many assumptions are made, which can be found in the assumptions and constraints table. As a high overview, the team assumes many things such as:

* A mobile application will be developed for residents.
* A web application will be developed for building administrators.
* A REST API will be developed as a middleman between the frontend applications and the database.
* A relational database will be used to store data.
* The project with be planned, analyzed, designed, and developed for free using a wide variety of technologies and programs. Programs like Figma will use the student plan to develop wireframes of the applications.
* Technologies will be chosen to develop this network. Proposed technologies have been listed in the risk management plan and proof of concepts will be developed before technologies have been officially chosen.

Other assumptions have been made that are initially out of scope for the project. Refer to the assumptions and constraints table for other assumptions.

**Proposed High Level Drawing**

Below is a high-level drawing of how different technologies will communicate with one another.

A diagram of a computer and mobile application

Description automatically generated

An admin will interact with the web application and a user (resident) will interact with the mobile application. When either an admin or user would like to make a request, the request will be sent to the REST API, which will communicate to the relational database. The response will be sent back up to the respective application. An example of this flow is when an admin verifies a user on the web application. The request will be sent to the API, which will then interact with the database to update the user in the database to verified. After the execution is complete, the API receives a response and will return the response back to the client application with either a success or failure.

**Solution**

Knockk is an extensive full stack project. Below is a diagram of the proposed technologies in the technology stack. Note that this drawing is the exact same as the high-level drawing, but proposed technologies have replaced generic icons.

A diagram of a application

Description automatically generated

The admin application’s proposed framework is React as it is a web application. The user facing frontend application will be a mobile application. Users may have different phones (Android, Apple, etc.), but the application should be compatible with iOS and Android, so React Native is the proposed solution because it allows for cross-platform development. Both applications will consume a REST API, which is proposed to be built with Express. The API will communicate to a relational database to store data. The proposed solution for the database is MySQL.

All proposed technologies will be thoroughly researched and will have proof of concepts on or before November 14, 2024, so that the project can continue moving forward. This is when final technical decisions will be made.

These are proposed solutions, pending research and development of proof of concepts. If React Native is not considered fit Flutter will be pursued; if React is not considered fit, Next.js or React Native (for web) will be pursued; if Express is not considered fit, Nest.js will be pursued; if MySQL is not considered it; PostgreSQL will be pursued.

**Proposed Technologies for Design and Development**

To make this project successful, different tools will be used to help the organize, develop, and execute the project. Below lists the proposed tools that will be used.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Design** | **Project Management** | **Code Management** | **Documentation** | **Diagram** |
|  |  |  |  |  |
| Figma | Jira | GitHub | Word | Draw.io |

Figma will be used for wireframing frontend applications. Jira will be used for work breakdown to plan and track tasks (project management). GitHub will be used for version control of code. Word will be used for documentation of any files related to this project. Lastly, Draw.io will be used for technical diagraming (like UML and ER diagrams). All these tools have been chosen because they are widely used in the industry.

Project Controls

To minimize risks turning into issues, the team has outlined risks with their impact, mitigation, and contingency plan in the table below. Throughout the project, risks will be mitigated as best they can, but if a risk becomes an event, it will be listed in the issue log. The team will work as best they can to minimize risks for the project's duration. As the project continues, risks will continually be added as they become known.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Risk Management | | | | |
|  | **Risk Probability** | **Risk Impact** |  |  |
| **Event Risk** | **(high, medium, low)** | **Risk Mitigation** | **Contingency Plan** |
| Figma starts incurring charges. | Low | The team would have to pay $20 a month. | The team will keep up to date with Figma policies and charges. | Word will be used for wireframing. |
| Azure starts incurring charges. | Low | Th team will have to pay the dues or lift and shift to another cloud provider. | The team will know how to lift and shift to AWS as well. | Lifting and shifting to a cloud provider is an out-of-scope capability. |
| GitHub starts incurring charges. | Low | The team would have to pay out of pocket. | The team will keep up to date with GitHub policies and charges. | Version control will be changed to Bitbucket. |
| Draw.io starts incurring charges. | Low | The team would have to pay out of pocket. | The team will keep up to date with Draw.io policies and charges. | Word will be used for diagraming. |
| The admin application is not fully functioning. | Low | Users will not be able to use the mobile application because the admin must verify the user before their account becomes activated. | Create the admin application at the first few stages of the application. | The project would be placed in jeopardy of completion. Minimal functionality will be completed will a basic UI design. |
| Cloud provider fails. | Low | While the uptime for cloud providers is high, it is never 100%. | Ensure all configurations are correct to keep the application up and running. | Lifting and shifting to a cloud provider is an out-of-scope capability. |
| The team has some experience with React. | Medium | The team needs to brush up on React | Research and create a proof of concept using React by 10/14/2024. | If React is not the best choice, Nest.js or React Native (for web) will be pursued. |
| The team has not decided on the specific tech stack. | Medium | Delays moving forward with the project. | Research and create a proof of concept of the proposed technologies so a decision can be made by 10/14/2024. | If proposed technologies are not the best fit, the contingency plan will be followed for backup technologies. |
| The team has little experience with React Native. | Medium | The team would not be able to build the front-end applications. | Research and create a proof of concept using React Native by 10/14/2024. | If React Native is not the best choice, Flutter will be pursued. |
| The team has little experience with Flutter. | Medium | The team would not be able to build the front-end applications. | Research and create a proof of concept using Flutter by 10/14/2024. | If Flutter is not the best choice, React Native will be pursued. |
| The team has little experience with Express. | Medium | The team would not be able to build an API; the intermediary of the frontend and backend. | Research and create a proof of concept using Express by 10/14/2024. | Nest.js will be researched and a proof of concept will be developed if Express is not fit. |
| The team has no experience with JWTs. | Medium | API will not be secured appropriately giving anyone access to the endpoints. | If all features in scope are completed before 5/3/2025, real-time databases will be researched, and proof of concept will be developed. | Initially out of scope. |
| The team has no experience with real-time databases. | Medium | Users will not be able to chat with one another. | If all features in scope are completed before 5/3/2025, real-time databases will be researched, and a proof of concept will be developed. | Initially out of scope. |
| The team has no knowledge of making the API public facing. | Medium | The API will only be used running locally if it is not public facing. | If all features in scope are completed before 5/3/2025, real-time databases will be researched, and a proof of concept will be developed. | Initially out of scope. |
| The team has no knowledge of building out a push notifications feature. | Medium | The app will not have a push notification feature. | If all features in scope are completed before 5/3/2025, push notifications will be researched, and a proof of concept will be developed. | Initially out of scope. |
| The team has no experience deploying to the App or Google Play stores. | Medium | The mobile application will not be deployed to the App or Google Play stores. | If all features in scope are completed before 5/3/2025, research will be done on how to deploy to the stores. | Initially out of scope |

There are no issues with the project, but in the future, issues that arise will be logged in the table below.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Issues Log | | | | | | | | |
| **ID** | **Description** | **Project Impact** | **Action Plan/Resolution** | **Owner** | **Importance** | **Date Entered** | **Date to Review** | **Date Resolved** |
| 1 |  |  |  |  |  |  |  |  |

The table below shows where changes to the project proposal will be tracked. Currently there are no changes because the project is currently in the initial draft stage.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Change Control Log | | | | | | | | | |
| **ID** | **Change Description** | **Priority** | **Originator** | **Date Entered** | **Date Assigned** | **Evaluator** | **Status** | **Date of Decision** | **Included in Rev. #** |
| 1 |  |  |  |  |  |  |  |  |  |

Project Cost and Schedule

This project will be developed free of cost (refer to risk management). For programs that typically require cost, student plans are utilized which offer the program for free.

Also refer to the work breakdown structure for the project schedule. For high level tasks, a Kanban board will be used, which can be viewed [here](https://seniorproject-graceradlund.atlassian.net/jira/software/projects/SPSD/boards/2/timeline). For clarity, epics are used for high level tasks (tasks in the work breakdown structure). For each epic, there will be a board which will contain lower-level tasks that break down the epic into smaller parts. Refer to the screenshot below which displays the board for this proposal.

A screenshot of a project management

Description automatically generated

All tasks have been moved to done because the project is complete. The board displays the current epic in progress, with the tasks associated. To view the board, click [here](https://seniorproject-graceradlund.atlassian.net/jira/software/projects/SPSD/boards/2).

Appendix A – References

Gee, D. (2021, March 3). *Know thy neighbor? Survey shows many don’t know their neighbors by name*. Retrieved from Wave: https://www.wave3.com/2021/03/04/know-thy-neighbor-survey-shows-many-dont-know-their-neighbors-by-name/

*New International Version*. (20011). Bible Gateway. https://www.biblegateway.com/versions/New-International-Version-NIV-Bible (Original work published 1973)

Appendix B – Copyright Compliance

Draw.io – Usage Terms - <https://www.drawio.com/doc/faq/usage-terms>

Draw.io is the proposed program that will be used for project diagramming. It is widely used at Grand Canyon University.

Express – Terms of Use - <https://images.prismic.io/openjsf/>

Express is the proposed framework for the development of a Rest API. It is a common framework used across the software industry.

Figma – Terms of Service - <https://www.figma.com/legal/tos/>

Figma is the proposed prototyping/wireframing program for front-end applications. The student version will be used for free. It is commonly used in the industry.

GitHub – Terms of Service - <https://docs.github.com/en/site-policy/github-terms/github-terms-of-service>

GitHub is the proposed program for version management. It will store all the repositories for the project. It is commonly used in the software industry.

Jira – Customer Agreement - <https://www.atlassian.com/legal/atlassian-customer-agreement#intro>

Jira is the proposed program for project management. It will break large tasks into smaller ones so that progress on the project can be viewed and organized. It is commonly used in the industry,

MySQL – Terms of Use - <https://www.oracle.com/legal/terms/>

MySQL is the proposed database management system. It is commonly used in the industry.

React and React Native – Terms of Use - <https://opensource.fb.com/legal/terms/>

Both React and React Native are developed by Meta. The two are proposed for the development of front-end applications. React will be used for the web application and React Native for the mobile application. They are both common frameworks in the industry.

Word – Terms of Use - <https://www.microsoft.com/en-us/legal/terms-of-use>

Word is the proposed program for project documentation. Files will be backed up on OneDrive so they can be accessed from anywhere. It is commonly used in the industry.