COLORADO SCHOOL OF MINES

EARTH • ENERGY • ENVIRONMENT

# CSCI 370 Final Report

**JO-3D**

Jonas Edlestein
Joseph Reedy
Johnny Bryant

Revised November 20, 2024

**A Multi-Dimensional Software Development Team**

CSCI 370 Fall 2024

Prof. Kathleen Kelly

Table 1: Revision history

| Revision | Date | Comments |
|---|---|---|
| New | Aug 20, 2024 | Completed Sections:<br>I. Introduction<br>II. Functional Requirements<br>III. Non-functional Requirements<br>IV. Risks<br>V. Definition of Done<br>XI. Team Profile<br>References<br>Appendix A – Key Terms |
| Rev – 2 | Sep 17, 2024 | Updated Sections<br>I. Introduction<br>II. Functional Requirements<br>III. Non-functional Requirements<br>Completed Sections:<br>VI. System Architecture |
| Rev – 3 | Oct 19, 2024 | Completed Sections:<br>VII. Software Test and Quality<br>VIII. Project Ethical Considerations |
| Rev - 4 | Nov 10, 2024 | Completed Sections:<br>IX. Project Completion Status<br>X. Future Work<br>XI. Lessons Learned |
| Rev –5 | Nov 20, 2024 | Finishing touches on all sections |
| Rev - 6 | Nov 28, 2024 | Completed Sections:<br>VII. Technical Design<br>Updated Sections:<br>All sections were updated with suggestions from peer review |
| Rev - 7 | Dec 2, 2024 | Added a “Peer Review Section to make submission easier<br>Completed Sections:<br>XV. Annotated Peer Reviews |

# Table of Contents

# I. Introduction

According to a study done by Robin, which offers a platform for desk booking, room scheduling, visitor management and workplace analytics, 40% of companies are currently utilizing half of their available space or less, with only 28% of businesses using 100% of their offices. Additionally, the demand for flexible work arrangements, including coworking spaces and short-term leases, is on the rise. According to a survey by Clutch, 60% of coworking space members in the U.S. are interested in flexible lease terms. This trend creates opportunities for businesses to offer their underutilized space for short-term leases to renters who cannot afford their own spaces or want to take advantage of the cost savings a coworking space offers. The purpose of this project is to create an app called "LeaseLinks" that will serve both space owners and space renters by limiting the effort it takes to link the two together. By doing so, space owners will have a tool to monetize previously underutilized space and space renters will have a tool to minimize their costs related to space rental. The ultimate goal of team JO-3D and the client Boss Ventures was to move the app to the MVP (Minimal Viable Product) stage. However, this was a lofty goal, and intermediate goals discussed later in this document were defined by the client. The client's end goal is to monetize the app for financial gain.

A previous Mines Software Development Team has already worked on the project. The project had not reached its MVP status. According to the client (Boss Ventures) the previous team had already started on the following:

- Homepage (Search Page)
- Login / Register New User
- Profile Creation – currently all information including images is **hard-coded**.

### *The client provided Tech Stack is as follows:*

**Back-end:**

- Firebase
- Firestore

**Version Control:**

- Github

**Front-End:**

- Vue.js
- Quasar
- Pinia

The client (Boss Ventures) team member, Mike Halverson, who is the team's lone developer/technical representative will be responsible for maintaining the software after the product is initially developed.

# II. Functional Requirements

The goal of this project is to create the MVP version of a web app designed to connect business property leases to companies/clients looking to rent spaces. Anywhere from renting an office building to renting a chair in a salon. The group before us left our project team with some building blocks so our functional requirements are based somewhat on what we have and the necessary next steps for the project. These requirements are listed below:

- **CRUD** (Create, Retrieve, Update, Delete) functionality for property listing
- **RBAC** (Role based access control)
    - Owners
        - edit, delete, create, and view listings
    - Renters
        - may only view listings
- Mobile Optimization – The application should render on all devices currently on the market. There should be no artwork, buttons, text, logos, or any other elements within the application that are not visible, collide, or are covered by other elements on the application due to screen size induced crowding.

# III. Non-Functional Requirements

- The web app should connect to a database that holds all user and other information
- The web app should protect users' information such as passwords

# IV. Risks

## Risk 1 – The risk of non-completion of the MVP due to lacking the skills required to get the job done.

- **Likelihood** (Unlikely, Likely, Very Likely)
  Team members currently possess most of the skills required to complete the MVP.
- **Impact (**Minor, Moderate, Major)
  Should the MVP not be completed, the client would not have a product to begin testing with users. Additionally, each moment they do not have a product to take to market, the door remains open for another company getting a similar product to market before they do.
- **Risk mitigation plan**
  Team members were committed to sharing their knowledge with other team members. Additionally, team members researched additional skills needed to complete the project.

## Risk 2 – The risk of breaking current project infrastructure or imposing extra work on client due to improper use of version control.

- **Likelihood** (Unlikely, Likely, Very Likely)
  **Impact** (Minor, Moderate, Major)
  Improper version control practices could lead to anything from inconveniencing the client with easily remedied merge conflicts to having to revert to older versions of the product. This could lead to the team creating extra work for our client and potentially lead to lost work and wasted time for the team.
- **Risk mitigation plan**
  Team members and client tech representative, Mike Halverson, have agreed upon a standardized operating procedure for all version control operations for the duration of the project.

# V. Definition of Done

- Have working features that align with the stakeholders needs
- Have an MVP ready to present

**NOTE:** The client does not expect the product to be done but is rather defining this phase of the project to be done if the following are implemented:

- **CRUD** (Create, Retrieve, Update, Delete) functionality for property listing
- **RBAC** (Role based access control) only owners can edit and delete listing
- Mobile Optimization – The application should render on all devices currently on the market. There should be no artwork, buttons, text, logos, or any other elements within the application that are not visible, collide, or are covered by other elements on the application due to screen size induced crowding.
- All additions and changes to the application are merged properly and delivered to the client on the agreed upon GitHub repository.

# VI. System Architecture

## 1. Front-End:

- **Vue.js**: The front-end of the application is built using Vue.js, a progressive JavaScript framework for building user interfaces. Vue.js handles the structure and behavior of the user interface.
- **Quasar:** Quasar is used as a UI framework on top of Vue.js. It provides a set of responsive, highly customizable components that allow for a consistent design and faster development.
- **Pinia**: Pinia is a state management library for Vue.js. It manages the application's global state, allowing different components to share and synchronize data.

## 2. Back-End:

- **Firebase**: Firebase is used for the back-end services. It provides a suite of cloud-based services that include authentication, real-time databases, cloud functions, and hosting. Firebase handles user authentication, file storage, and other back-end services.
- **Firestore:** Firestore, a NoSQL document database, is part of Firebase and is used for storing and syncing data in real time. It allows for efficient querying and is optimized for low-latency mobile and web applications.

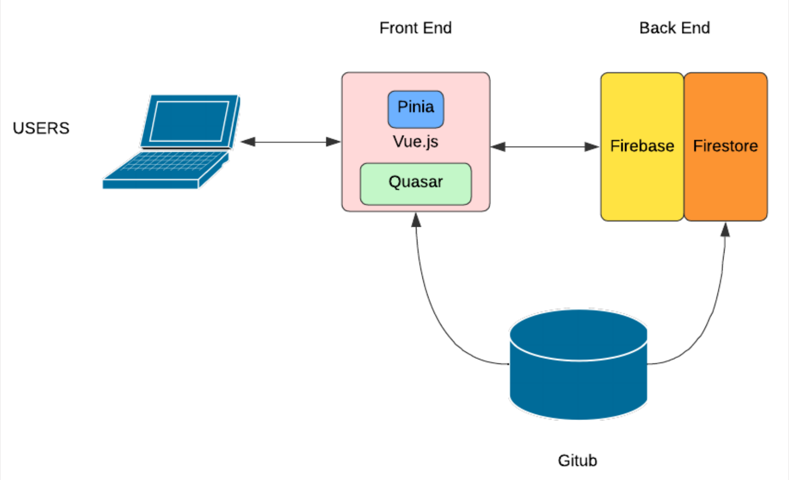
## 3. Version Control:

- **GitHub**: GitHub is used for version control and collaboration. The codebase, including both front-end and back-end components, is managed and versioned through GitHub repositories. This allows for collaborative development, with multiple developers contributing to the project.

## Example of interaction of System Architecture Components

- A user lands on the landing page of the application controlled by Vue.js
- The user registers as a user and their credentials are stored in the database Firebase
- A user wants to add a listing of their property, so they navigate to a page rendered by file PageAddListing.vue which is a Vue.js file. PageAddListing.vue controls the front-end visual rendering and functionality of the application page and has access to functions that access data stored in the backend in Firebase.
- The user enters their data which is stored in the backend database Firebase
- Github is used by the developers of the application to work remotely and simutaneously on the application. The latest “version” of the application is centrally stored on a remote repository that all the developers can update and attain whenever needed.

## System Architecture Visualization



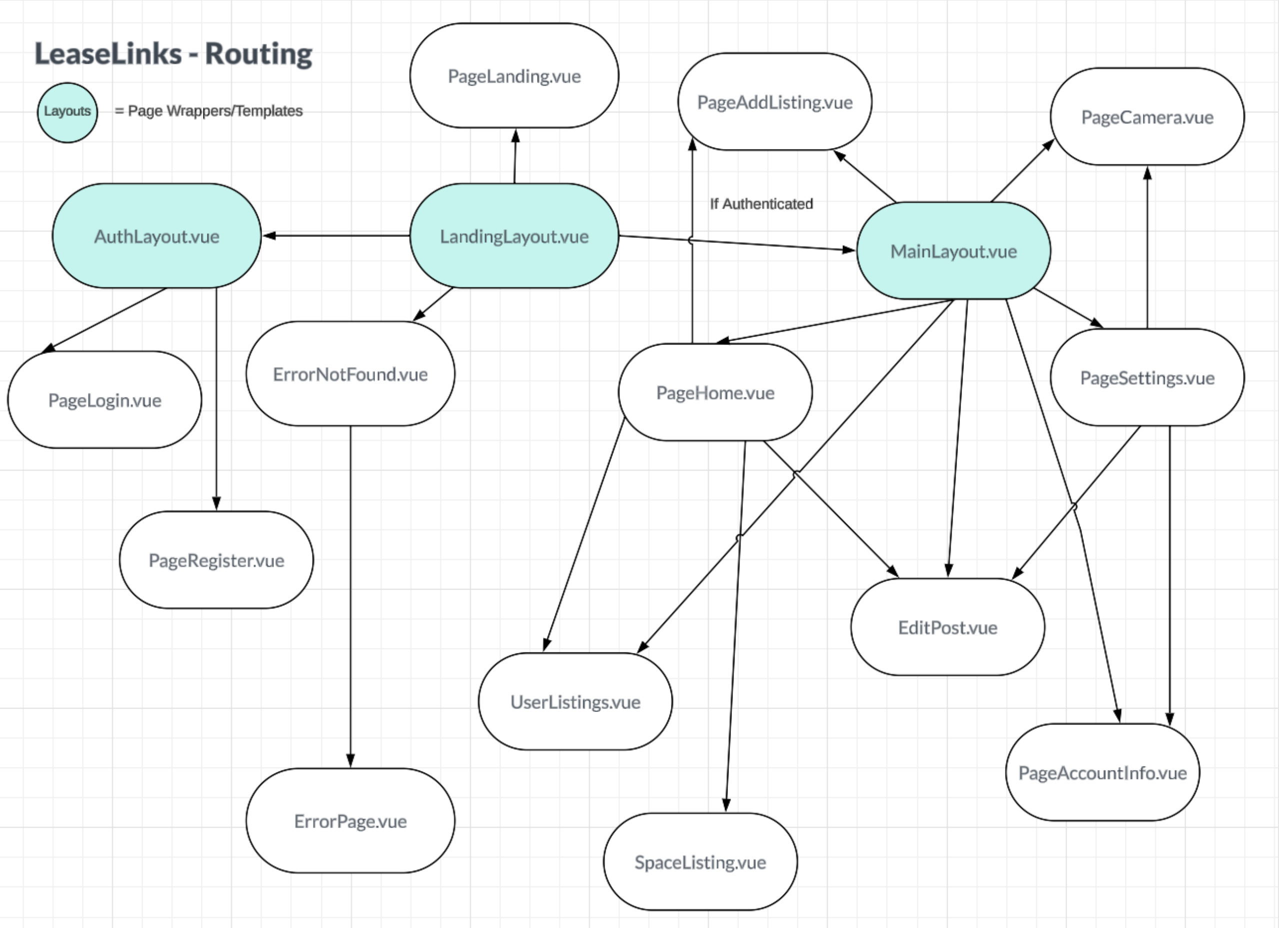
## Data Flow and Interaction:

- **User Interaction**: Users interact with the application through the front-end, built with Vue.js and Quasar. The UI components fetch, and display data stored in Firestore and managed by Pinia for state management.
- **State Management**: Pinia ensures that the application's state is consistent and synchronized across various components. This allows for a smooth user experience with real-time updates.
- **Back-End Communication**: The front-end communicates with Firebase to handle user authentication, data storage, and other back-end services. Firebase uses Firestore to store user data, which is then fetched and displayed on the front-end.
- **Version Control**: GitHub tracks changes to the codebase, allowing developers to manage updates, collaborate, and deploy new features or fixes.

# VII. Technical Design

## Routing

The following figure is a flow chart detailing the routing paths on the LeaseLinks application:



Note the layout files highlighted in light blue. In a Vue.js and Quasar application, layouts are not standalone pages but wrappers/templates that define the overall structure for the pages rendered inside them. Layouts provide reusable structural components like headers, footers, sidebars, and navigation bars that maintain consistency across multiple pages.

## Firebase (Database) – Custom Built Functional Modules

In the LeaseLinks application, .js files were intentionally designed as internal utility modules in Vue.js to centralize commonly used functionality. This centralization was a core aspect of our technical design, aimed at streamlining tasks like creating, retrieving, updating, or deleting data in Firebase. By consolidating these operations into reusable modules, the application ensures consistency, maintainability, and scalability. Below is a list of the modules, their purposes, and where they are used. This was given to our client as a "handoff" tool used for future developers on this project. Some modules are not currently in use but have been implemented as part of the forward-looking design to support future features, such as user email verification and other planned enhancements.

### Creating Posts/Listings

- **firebase-createPosts.js**
  - **Purpose**
    - It provides a backend interface for creating and saving posts/listings.
    - Connects the frontend (e.g., PageAddListing.vue page) to the Firestore database.
  - **Use**
    - Ensures posts are tied to the currently authenticated user (uid), enabling features like filtering posts by user.

### Retrieving All Posts/Listings

- **firebase-getPosts.js**
  - **Purpose**
    - Retrieves all posts/listings from the "posts" collection in Firestore.
    - Parses each document in the snapshot and creates an array of all posts, including their Firestore document IDs (id) and their data.
    - Returns this array for use in components that need to display multiple posts.
  - **Use**

    - Displays all the current site user posts/listings on PageHome.vue page.

## Retrieving One User Post/Listing

- firebase-getListing.vue
  - **Purpose**
    - Retrieves a single user post/listing by its unique listingId.
  - **Use**
    - Currently not used.

## Delete One User Post/Listing

- **firebase-deleteListing.js**
  - **Purpose**
    - Deletes a single user post/listing by its unique listingId.
  - **Use**
    - Deletes user post/listing on UserListing.vue page

## Retrieve All User Posts/Listings

- **firebase-getUserListings.js**
  - **Purpose**
    - Retrieves all user posts/listings associated with the current authenticated user's identification (uid).
  - **Use**
    - Used to retrieve and display all user posts/listing on UserListings.vue page.

## Update an Existing Post

- **firebase-updatePost.js**
  - **Purpose**
    - Updates an existing post in the posts collection of Firestore with new data (updatedData) based on the provided listingId.
  - **Use**
    - Used when a user updates their post/listing on EditPost.vue page.

## Retrieve User Information

- **firebase-getUserInfo.js**
  - **Purpose**
    - Retrieves user data from Firebase Authentication based on a user's unique ID (uid).
  - **Functionality**
    - **Provides two functions:**
      - getListingUser(uid):
        - Fetches a user's complete authentication record using getAuth().
      - getUserEmail(uid):
        - Retrieves the email address associated with a user's uid using the Firebase Admin SDK.
  - **Use**
    - Currently Not used

## Save User Data to Firestore

- **firebase-userdata.js**
  - **Purpose**
    - Saves additional user profile details (e.g., name, phone number, business name, location) into Firestore under the user's collection.
  - **Functionality**
    - Saves data submitted via forms (e.g., PageAccountInfo.vue page) into a Firestore document located at users/<uid>.
    - Fields include:
    - Full name
    - Business name
    - Phone number
    - Location
    - Owner
    - Renter
    - Etc.

        - **Use**
            - Used to store custom user information beyond what is available in Firebase Authentication.
            - Supports features like profile pages, where extended user details are displayed.

## Centralize Initialization of Database and Authentication

- **index.js**
    - Purpose
        - Centralizes the setup and initialization of Firebase services (e.g., Authentication, Firestore).
        - Provides utility functions (getListing and getUserEmail) to interact with Firestore.
    - **Key Responsibilities**
        - Initializes Firebase with your app's configuration (firebaseConfig).
        - Sets up Firebase services like:
            - Authentication (auth)
            - Firestore Database (db)
        - Uses onAuthStateChanged to track user authentication state changes and update the userStore and LocalStorage.
        - Provides helper functions:
            - getListing(listingId): Fetches a specific listing from Firestore.
            - getUserEmail(uid): Retrieves a user's email from the user's collection in Firestore.
    - **Why It's Necessary**
        - Without this file, you would need to repeat Firebase initialization code and utility functions in every component or module where Firebase is used, leading to duplication and maintenance issues.

## Register a New User

- **firebase-register.js**
    - **Purpose**
        - Creates a new user in Firebase Authentication using an email and password.
        - Updates the user's profile with a display name based on the provided first and last name.
    - **Use**
        - Used on the PageRegister.vue page when a user submits a registration form.
        - Handles the creation of a new user account and sets up their profile information.

## User Login

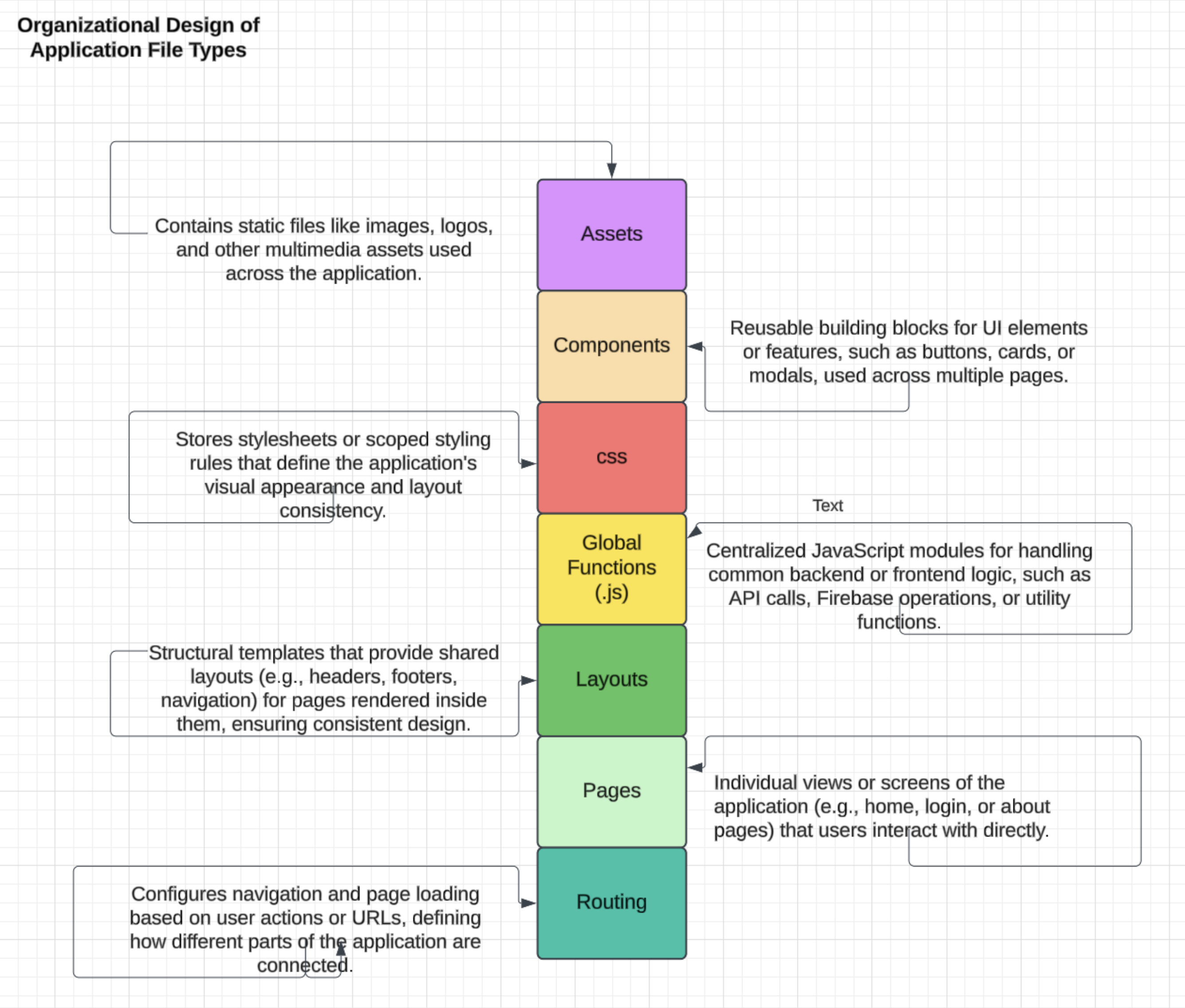
- **firebase-login.js**
    - **Purpose**
        - Logs in a user via Firebase Authentication using their email and password.
        - Updates the application state (userStore) with the authenticated user's data on successful login or clears it on failure.
    - **Use**
        - Used on the PageLogin.vue page when a user submits their login credentials.
        - Handles session management and error notifications for login attempts.

## Sign Out a User

- **firebase-signout.js**
    - **Purpose**
        - Signs out the currently authenticated user using Firebase Authentication.
        - Clears the user's data from the application state (userStore) and provides feedback to the user during the process.
    - **Use**
        - Used in components or pages where a logout option is available, such as a settings page, profile dropdown, or navigation drawer.
        - Ensures the user is signed out of Firebase and their session is terminated.

## Organizational Design of File Types and Functions

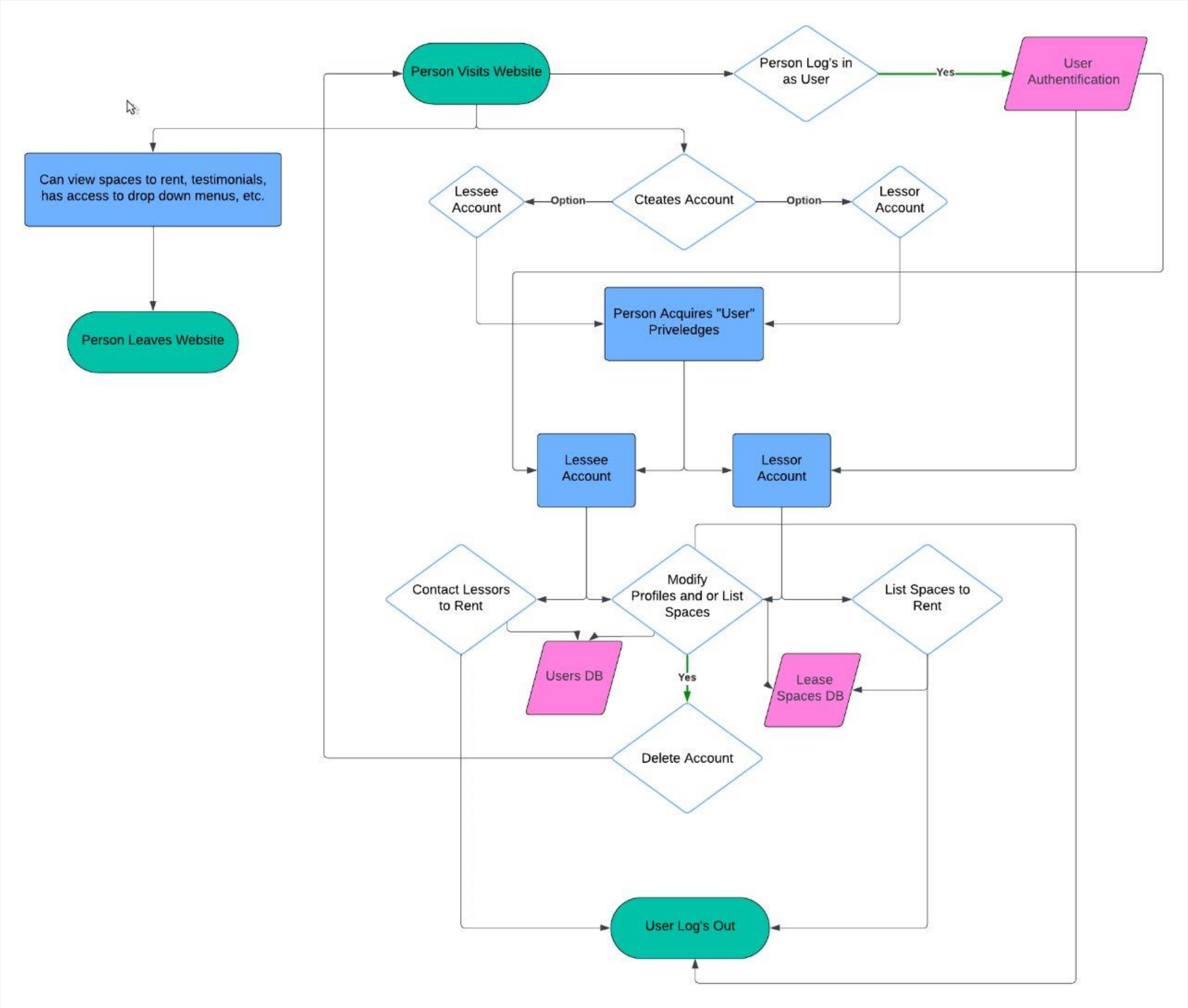
The following is the organizational design of file types according to their function. This was part of the handoff package given to our client as a tool to help future developers on their project adhere to the same design.



Organizational Design of Application File Types
Contains static files like images, logos, and other multimedia assets used across the application.
Assets
Components
Reusable building blocks for UI elements or features, such as buttons, cards, or modals, used across multiple pages.
Stores stylesheets or scoped styling rules that define the application's visual appearance and layout consistency.
css
Text
Global Functions (.js)
Centralized JavaScript modules for handling common backend or frontend logic, such as API calls, Firebase operations, or utility functions.
Structural templates that provide shared layouts (e.g., headers, footers, navigation) for pages rendered inside them, ensuring consistent design.
Layouts
Pages
Individual views or screens of the application (e.g., home, login, or about pages) that users interact with directly.
Configures navigation and page loading based on user actions or URLs, defining how different parts of the application are connected.
Routing

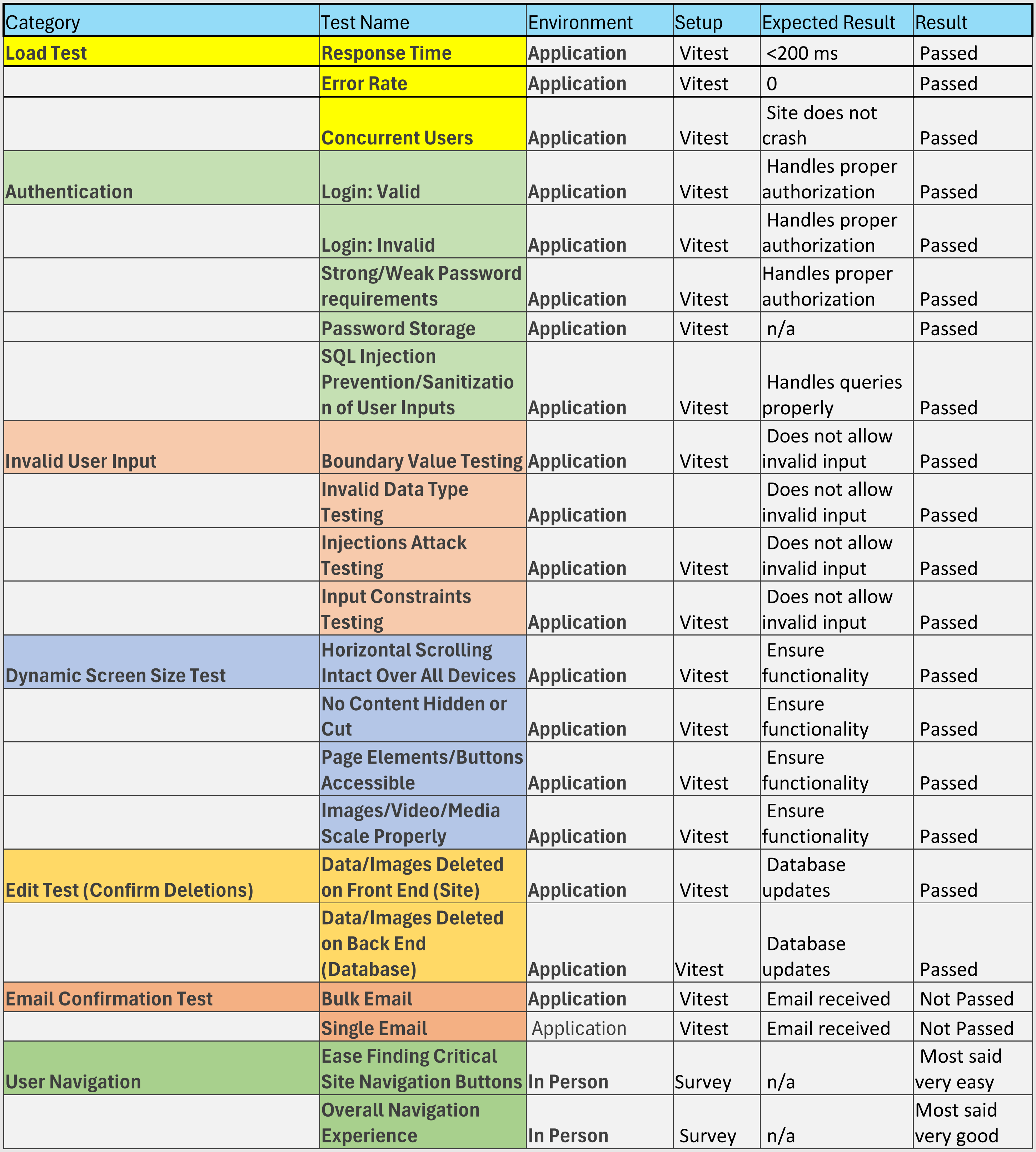
## RBAC (Role-Based Access Control) Design

The following was our team’s RBAC design for the application. However, due to time constraints we were not able to implement it. This design was passed off to our client for future groups to implement.



## VIII. Software Test and Quality

Team JO-3D implemented and provided the results for the following tests to the client, BossVentures in order to ensure the product that they are receiving is able to handle the tasks it was asked to perform. A more detailed description of each test is listed below the following spreadsheet:



| Category | Test Name | Environment | Setup | Expected Result | Result |
|---|---|---|---|---|---|
| Load Test | Response Time | Application | Vitest | <200 ms | Passed |
|  | Error Rate | Application | Vitest | 0 | Passed |
|  | Concurrent Users | Application | Vitest | Site does not crash | Passed |
| Authentication | Login: Valid | Application | Vitest | Handles proper authorization | Passed |
|  | Login: Invalid | Application | Vitest | Handles proper authorization | Passed |
|  | Strong/Weak Password requirements | Application | Vitest | Handles proper authorization | Passed |
|  | Password Storage | Application | Vitest | n/a | Passed |
|  | SQL Injection Prevention/Sanitization of User Inputs | Application | Vitest | Handles queries properly | Passed |
| Invalid User Input | Boundary Value Testing | Application | Vitest | Does not allow invalid input | Passed |
|  | Invalid Data Type Testing | Application |  | Does not allow invalid input | Passed |
|  | Injections Attack Testing | Application | Vitest | Does not allow invalid input | Passed |
|  | Input Constraints Testing | Application | Vitest | Does not allow invalid input | Passed |
| Dynamic Screen Size Test | Horizontal Scrolling Intact Over All Devices | Application | Vitest | Ensure functionality | Passed |
|  | No Content Hidden or Cut | Application | Vitest | Ensure functionality | Passed |
|  | Page Elements/Buttons Accessible | Application | Vitest | Ensure functionality | Passed |
|  | Images/Video/Media Scale Properly | Application | Vitest | Ensure functionality | Passed |
| Edit Test (Confirm Deletions) | Data/Images Deleted on Front End (Site) | Application | Vitest | Database updates | Passed |
|  | Data/Images Deleted on Back End (Database) | Application | Vitest | Database updates | Passed |
| Email Confirmation Test | Bulk Email | Application | Vitest | Email received | Not Passed |
|  | Single Email | Application | Vitest | Email received | Not Passed |
| User Navigation | Ease Finding Critical Site Navigation Buttons | In Person | Survey | n/a | Most said very easy |
|  | Overall Navigation Experience | In Person | Survey | n/a | Most said very good |

## Load Test

A **load test** is a performance test used to evaluate how an application behaves under normal or peak usage conditions. The goal is to determine how well the system performs when multiple users access it simultaneously, helping to identify bottlenecks and ensure the application can handle expected traffic. J0-3D will be testing the LeaseLinks site for the following:

- **Response Time** - How long it takes to process a request (e.g., page loads, API calls).
- **Error Rate** - Percentage of failed requests or errors under load.

- **Concurrent Users** - The number of users the system can support simultaneously.

## Authentication

Authentication is a critical part of any application's integrity and security both from a functional and security standpoint. JO-3D will test the following on the LeaseLinks application:

## Authentication - Functional Tests

- **Login with valid credentials** - Verify users can log in with correct usernames/emails and passwords.
- **Login with invalid credentials** - Test login attempts with incorrect passwords or non-existent users.

## Authentication - Security Tests

- **Password policies** - Check if the system enforces strong passwords (e.g., minimum length, special characters).
- **SQL Injection in login** - Verify the application sanitizes user inputs to prevent SQL injection attacks.
- **Password storage** - Check if passwords are securely hashed before being stored.

## Invalid User Input

Often users input incorrect information into fields on applications whether innocently or maliciously in site attacks. JO-3D will test for these scenarios including edge cases. Some of the things JO-3D will be testing for are listed below:

## Boundary Value Testing

- Input size limits: Try entering:
  - Too short: Empty fields or single characters.
  - Too long: Inputs exceeding the maximum allowed length (e.g., 51 characters in a 50-character field).
- Whitespace characters: Use spaces, tabs, or newlines as inputs (e.g., " ").

**Example:**

- Username: Enter 256+ characters (when only 255 allowed).
- Password**:** Leave the field empty.

**Expected Result:**

- Errors with clear messages (e.g., "Input exceeds maximum length").
- Application should not crash.

## Invalid Data Type Testing

- Enter numbers in text fields (e.g., username = 12345).
- Enter letters in numeric fields (e.g., age = abc).
- Using special symbols where not allowed (e.g., username = #@!user).

**Example:**

- Phone number: Enter abc123.

**Expected Result:**

- Should reject invalid data types (e.g., "Enter only numeric values").

## Injection Attack Testing

- SQL Injection: Enter SQL commands like:
  - ' OR 1=1; --
- XSS (Cross-Site Scripting): Enter:
  - <script>alert('XSS');</script>
- Command Injection: Try:
  - ; rm -rf /

**Expected Result:**

- Input should be sanitized or escaped (see Appendix) to prevent SQL attacks.
- No sensitive information leaked in error messages.

## Input Constraints Testing

- Password validation: Use inputs without special characters or required elements (e.g., password = password123 when special symbols are required).
- Email validation: Enter improperly formatted emails like user@@domain.com or user.com.

- Username validation: Use reserved words (e.g., admin, root).

**Expected Result:**

- Display relevant validation errors.
- Passwords should meet all password complexity rules.

## Dynamic Screen Size Test

**Dynamic screen** size testing ensures that an application provides a smooth, consistent, and responsive user experience across multiple devices and screen sizes, including desktops, tablets, and smartphones. JO-3D will utilize either Chrome's device emulator or real devices to ensure the LeaseLinks application is optimized on all known devices. JO-3D will check that on every device the LeaseLinks application does the following:

- The layout adapts without horizontal scrolling.
- No overlaps or content cuts on any screen size.
- Page elements (buttons, text) remain accessible.
- Image, video, and media scale properly.

## Edit Test (Confirm Deletions)

Proper data maintenance is critical for both the security and functionality of an application. JO-3D will test to ensure any data including images is properly deleted from the LeaseLinks application. JO-3D will ensure deletions performed will be deleted on both the front end (site) and the back end (database) of the application.

## Email Confirmation Test

JO-3D will test whether emails that are sent are received by the intended recipient. Testing will include the following:

- Bulk emails sent are received by all recipients and only one email is received by each recipient.
- Single emails are received by the proper recipient and all data in the sent email is in the received email.

## User Navigation

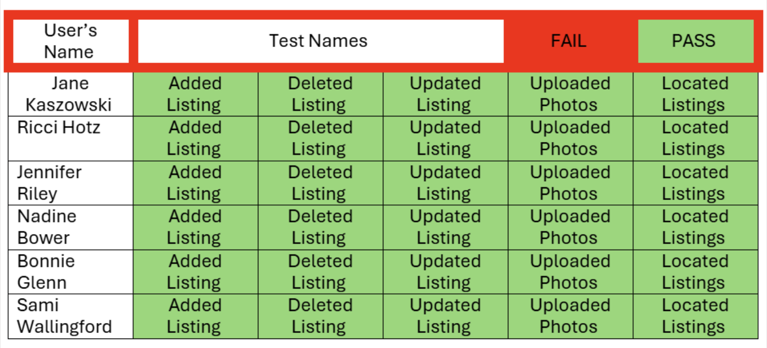
A positive user experience is critical for an application's success. JO-3D will have users test the navigability of the LeaseLinks application. Things that users will be asked to provide feedback on are:

- Ease to find critical site navigation action buttons.
- Overall experience navigating the application.

### *User Navigation Testing - Results*

Six human users helped test the usability of the application. Test outcomes were either "fail" or "pass." The tests conducted were whether the user was able to:

I. Add/create a listing
II. Delete a listing
III. Update a listing
IV. Upload photos
V. Locate their listings (and other people's listings)



| User's Name | Test Names | | | FAIL | PASS |
|---|---|---|---|---|---|
| Jane Kaszowski | Added Listing | Deleted Listing | Updated Listing | Uploaded Photos | Located Listings |
| Ricci Hotz | Added Listing | Deleted Listing | Updated Listing | Uploaded Photos | Located Listings |
| Jennifer Riley | Added Listing | Deleted Listing | Updated Listing | Uploaded Photos | Located Listings |
| Nadine Bower | Added Listing | Deleted Listing | Updated Listing | Uploaded Photos | Located Listings |
| Bonnie Glenn | Added Listing | Deleted Listing | Updated Listing | Uploaded Photos | Located Listings |
| Sami Wallingford | Added Listing | Deleted Listing | Updated Listing | Uploaded Photos | Located Listings |

# IX. Project Ethical Considerations

JO-3D is conscious of the many ethical considerations that must be taken during implementation of modern technological projects. JO-3D has divided those considerations into the following two categories:

- Techno-Professional Ethical Considerations
- Techno-Social Ethical Considerations

These moral and ethical considerations are founded on team JO-3D's personal high moral standards, the belief in a social contract, and standards set forth by the IEEE (Institute of Electrical and Electronics Engineers) and ACM (Association for Computing Machinery), two prominent professional organizations in the field of computer science and engineering. When relevant the relevant ACM/IEEE principal reference number will be listed with the corresponding ethical consideration for this project.

## Techno-Professional Ethical Considerations

### Data Confidentiality and Integrity

IEEE, 3.12. Work to develop software and related documents that respect the privacy of those who will be affected by that software. [2]

- JO-3D will work to ensure all user data on the LeaseLinks application is secure and free from malicious interventions like Cross-Site Scripting attacks. JO-3D is aware that a breach of user data could have a ripple effect since users often use data like passwords on multiple sites and applications. JO-3D will utilize software development products that have built in tools to prevent malicious attacks on the site both external and internal to ensure user's data is protected. Additionally, JO-3D will test the efficacy of said products. When weighing the efficacy of systems employed to protect users' data JO-3D uses the "reversibility" test defined below.

  **Reversibility test:** would this choice still look good if I traded places? (i.e., if I were one of those adversely affected by it?)

### Quality of Product

- IEEE, 3.08: Ensure that specifications for software on which they work have been well documented, satisfy the users requirements, and have the appropriate approvals. [2]
- IEEE, 3.10: Ensure adequate testing, debugging, and review of software and related documents on which they work. [2]

JO-3D always employs the best industry practices in all things related to software development. JO-3D will continually touch base with the client/user to ensure all of their specifications and requirements are met. Additionally, JO-3D will perform extensive testing to ensure any edge cases not addressed or foreseen by the client/user are properly addressed. JO-3D is aware that if these measures to ensure product quality are not properly implemented the client/user will be left with a product that does not meet their standards and may be a product that ends up costing them revenue without any foreseeable return.

## Social Ethical Considerations

### Project Confidentiality and Integrity

- ACM, 1.3: Be honest and trustworthy. [1]
- ACM, 2.03: Use the property of a client or employer only in ways properly authorized, and with the client's or employer's knowledge and consent. [1]

All members of team JO-3D have entered into a non-disclosure agreement with Boss Ventures and must consider the implications of not adhering to said agreement. In doing so, the client at a minimum will lose revenue and lose the opportunity to bring a novel product to the market.

### Technological Development Creates/Destroys Jobs

- ACM, 3.1: Ensure that the public good is the central concern during all professional computing work. [1]
- ACM, 1.2: Avoid Harm. [1]

Team JO recognizes the delicate balance of technological development and job opportunities. Team JO-3Drecognizes that the development of the app LeaseLinks may have a causal effect of jobs like leasing agents to become obsolete. Team JO-3D hopes that with the development of this application more lives are made better. Additionally, team JO hopes the convenience and cost saving function the application provides outweighs any damage done within the job market.

### Mindful Professionalism

- IEEE, 2.1: Strive to achieve high quality in both the processes and products of professional work. [2]
- IEEE, 2.2: Maintain high standards of professional competence, conduct, and ethical practice. [2]

Team JO-3D recognizes that beyond project considerations the humans bringing the project to fruition must also be considered. Team JO-3D commits to positive, edifying, respectful, and uplifting human interactions with all involved on this project. JO-3D asks that each team member employ the "mirror", and "common practice" tests defined below when weighing whether an interaction with a client or team member is appropriate or beneficial.

- **Mirror test**: Would I feel proud of myself when I look into the mirror?
- **Common practice test**: What if everyone behaved this way?

## X. Project Completion Status

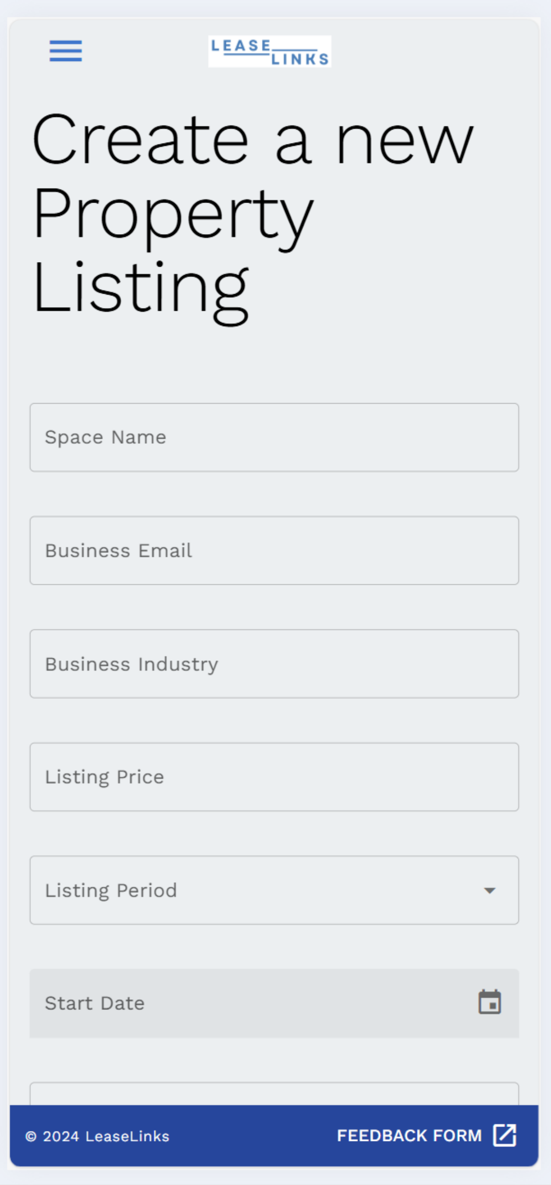
The client does not expect the application to be a minimal viable product by the end of the semester. Their focus is on our learning and making honest progress towards their goal of an MVP. Intermediate completion goals that were set were the implementation of 3 things:

1) CRUD (create, read, update, delete) for listings
2) Mobile optimization
3) RBAC (role-based access control) for lessors and lessees (stretch goal)

   a) CRUD and Mobile optimization were implemented. However, we were not able to get to RBAC due to a lack of time.

## CRUD (create, read, update, delete) for listings

### Create



Create a new Property Listing

Space Name

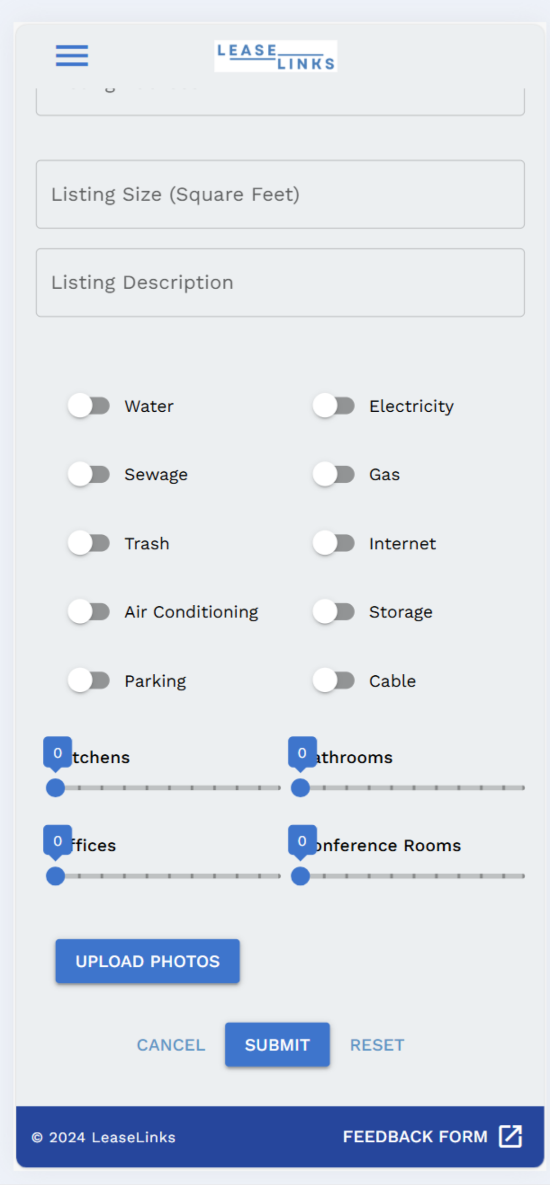
Business Email

Business Industry

Listing Price

Listing Period

Start Date



Listing Size (Square Feet)

Listing Description

Water ○ Electricity ○ Sewage ○ Gas ○ Trash ○ Internet ○ Air Conditioning ○ Storage ○ Parking ○ Cable ○

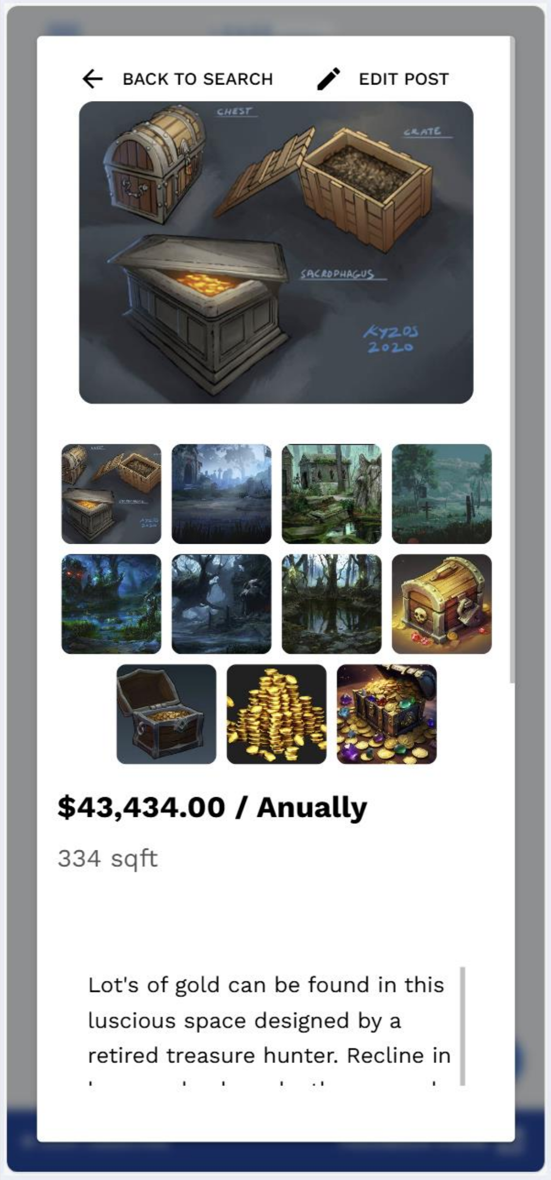
0 itchens

0 athrooms

0 ffices

0 onference Rooms

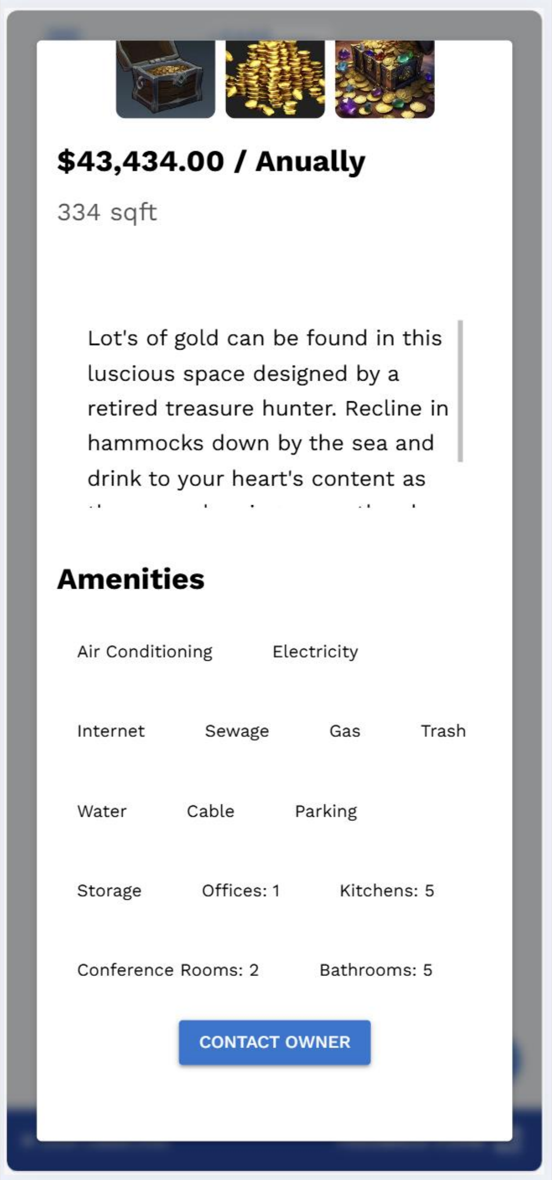
## Read



**$43,434.00 / Anually**

334 sqft

Lot's of gold can be found in this luscious space designed by a retired treasure hunter. Recline in



**$43,434.00 / Anually**

334 sqft

Lot's of gold can be found in this luscious space designed by a retired treasure hunter. Recline in hammocks down by the sea and drink to your heart's content as

**Amenities**

Air Conditioning Electricity

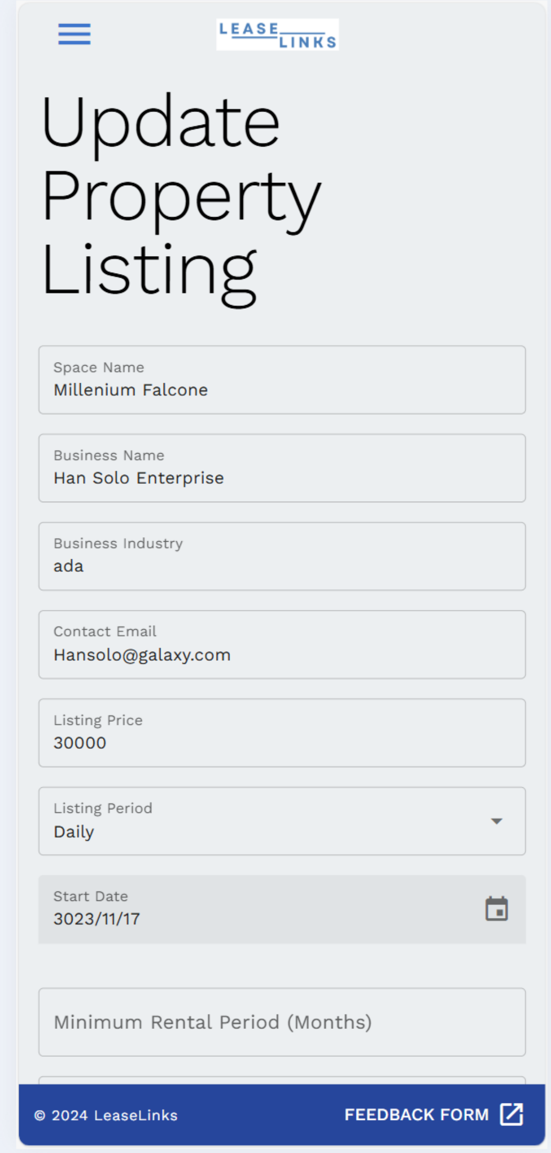
Internet Sewage Gas Trash

Water Cable Parking

Storage Offices: 1 Kitchens: 5

Conference Rooms: 2 Bathrooms: 5

## Update



# Update Property Listing

Space Name
Millenium Falcone

Business Name
Han Solo Enterprise

Business Industry
ada

Contact Email
Hansolo@galaxy.com

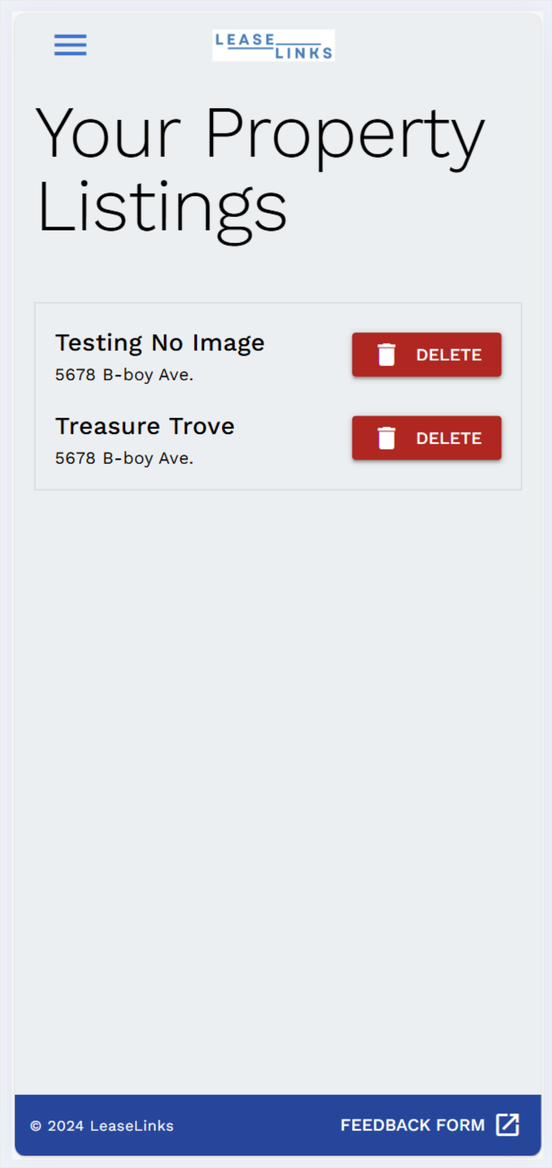
Listing Price
30000

Listing Period
Daily

Start Date
3023/11/17

Minimum Rental Period (Months)

### Delete



Your Property Listings

Testing No Image
5678 B-boy Ave.

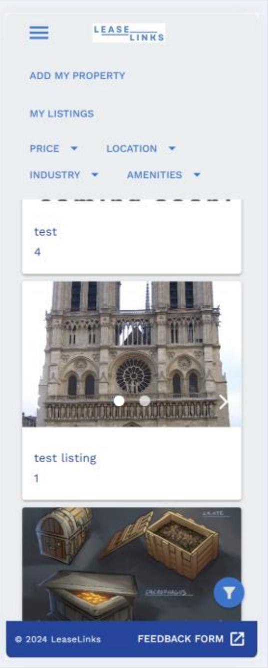
Treasure Trove
5678 B-boy Ave.

© 2024 LeaseLinks

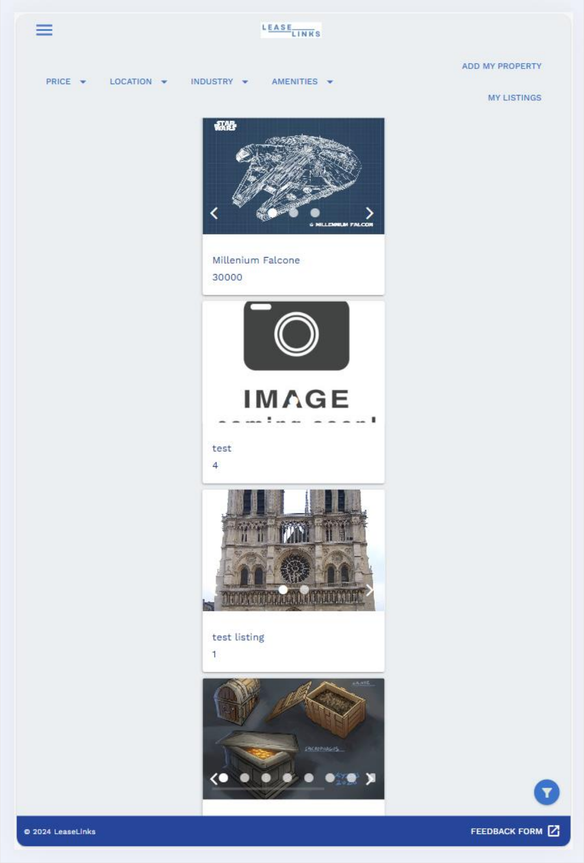
## Mobile optimization

Viewing of the application was made optimal for all devices ranging from 344 – 1280 pixels in height to 600 – 1368 pixels in width. For example, here is the home page for the smallest and largest height and width devices in the range provided:

*Galaxy Z fold 5 – 344 x 882*



*Surface Pro 7 – 912 x 1368*



# XI. Future Work

Future work on this web application involves several enhancements to improve usability, functionality, and access control. A primary feature yet to be implemented is a "Map View", which would allow users to see all available listings on an interactive map. This map would display location-based pins for each listing, giving users a spatial overview of available properties and helping them quickly identify options within specific areas. In terms of search functionality, the application currently provides labels to filter results by amenities, location, and other variables, but these filters are not yet connected to the backend. Implementing dynamic search capabilities would empower users to refine their search results effectively, finding listings that best meet their needs and preferences. Another essential future feature is the introduction of Role-Based Access Control (RBAC), which would allow different types of users—such as lessees and lessors—to have tailored access permissions. With RBAC, users could be limited to certain actions based on their roles, such as creating and editing only the listings they have posted. Future work implementations are summarized below in order of their respective priority:

## User RBAC

- Role Based Access Control (RBAC) for Lessee or Leasor to change the view and access to things like creating and editing posts based on the listings they have put up.

## Search Functions

- Currently there are labels to narrow down search based on amenities and location and other variables, but these menus do not have functionality on the backend.

## Map View

- Map based on listing addresses to see the listings available in a rendered map view with pins based on location

Additional future work might include enhancing user experience through notification systems, such as alerts for new listings or updates to saved properties, as well as introducing messaging functionality for users to communicate directly within the platform. Another potential improvement is integrating advanced analytics for property owners, providing them with insights into views, and other user interactions with their listings. These are features and updates that could help elevate the user experience of the app after the initial launch and testing has begun.

# XII. Lessons Learned

Overall, team JO-3D performed exceptionally well. However, like with all projects there were the inevitable bumps in the road and lessons that were learned. These lessons are listed below:

## Cache Staleness

During the development of our application, we encountered an issue known as "cache staleness" or "cache retention." This occurs when outdated versions of files are stored in the cache, causing older versions of the app to load instead of the latest updates. In our case, this resulted in the home page failing to load, leading one developer to mistakenly assume they had broken the app. As a result, they reverted to an older version via Git, which led to a loss of ten hours of work. After discovering the impact of cache staleness, our team now clears the cache each time before launching the application to ensure the most recent version is displayed.

## Thorough Assessment of Previous Development Work:

Team JO-3D initially believed we had thoroughly assessed prior development work before moving forward. However, since parts of the tech stack were new to some team members, there were gaps in understanding that led to redundant code. This caused confusion, especially with routing, where duplicate routing files existed. Some pages were routed by function and variables in one file, while others were routed in the duplicate, leading to inconsistencies. All redundant routing files were deleted. Going forward, we plan to review prior code more carefully to ensure a unified approach.

# XIII. Acknowledgments

We would like to give a massive shoutout to BossVentures LLC for helping foster our growth in web app development and for allowing us to be a small part of the exciting growth of their company. Special shoutout to Julianna Bologa and Mike Halverson who were our client liaisons throughout the project. You guys are/were wonderful, and your guidance was/is greatly appreciated! Lastly, we would also like to give a massive shoutout to Kathleen Kelly, our Advisor/Mentor/Idol on the school side.  Professor Kelly helped us to stay on track, achieve our goals, return a quality product to our client, and taught us a lot about teamwork and communication along the way.

# XIV. Team Profile

## Michael (Johnny) Bryant

Senior

Computer Science
Hometown: Los Angeles, California
Work Experience: US Army – Signal Support Systems Specialist Airborne, Fiber Optic Technologies – Project Foreman
Intel – Project Manager, Paragon Ballroom and Dance Instruction – Owner, Dance Instructor, Choreographer
*I am excited to be part of a project that not only allows me to apply some of what I have learned about Computer Science so far but also allows me to learn even more.*

## Jonas Edelstein

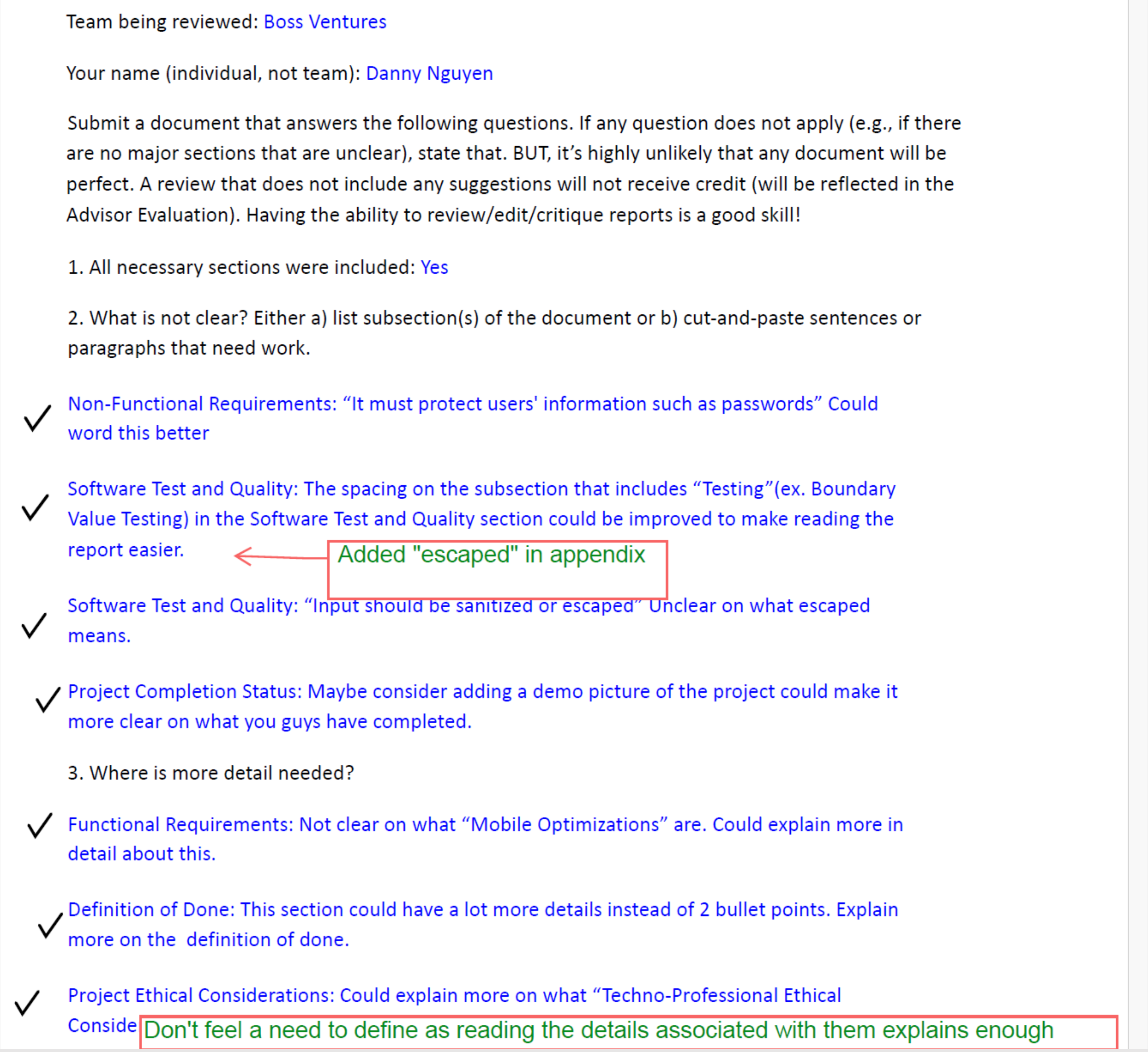
Senior
Computer Science
Hometown: Seattle, Washinton
Work Experience: IT Intern – Infinity Concrete construction, IT Intern – Lavner Education, Athletic Communications Associate – Colorado School of Mines.

## Joseph Reedy

Junior
Computer Science at Mines
Work Experience: Junior Software Engineer at Pax8
As a computer science major, I am excited to apply my knowledge and learn new skills.

# XV. Annotated Peer Reviews

## Review 1



Team being reviewed: Boss Ventures

Your name (individual, not team): Danny Nguyen

Submit a document that answers the following questions. If any question does not apply (e.g., if there are no major sections that are unclear), state that. BUT, it’s highly unlikely that any document will be perfect. A review that does not include any suggestions will not receive credit (will be reflected in the Advisor Evaluation). Having the ability to review/edit/critique reports is a good skill!

1. All necessary sections were included: Yes

2. What is not clear? Either a) list subsection(s) of the document or b) cut-and-paste sentences or paragraphs that need work.

✓ Non-Functional Requirements: “It must protect users' information such as passwords” Could word this better

✓ Software Test and Quality: The spacing on the subsection that includes “Testing”(ex. Boundary Value Testing) in the Software Test and Quality section could be improved to make reading the report easier.

Added "escaped" in appendix

✓ Software Test and Quality: “Input should be sanitized or escaped” Unclear on what escaped means.

✓ Project Completion Status: Maybe consider adding a demo picture of the project could make it more clear on what you guys have completed.

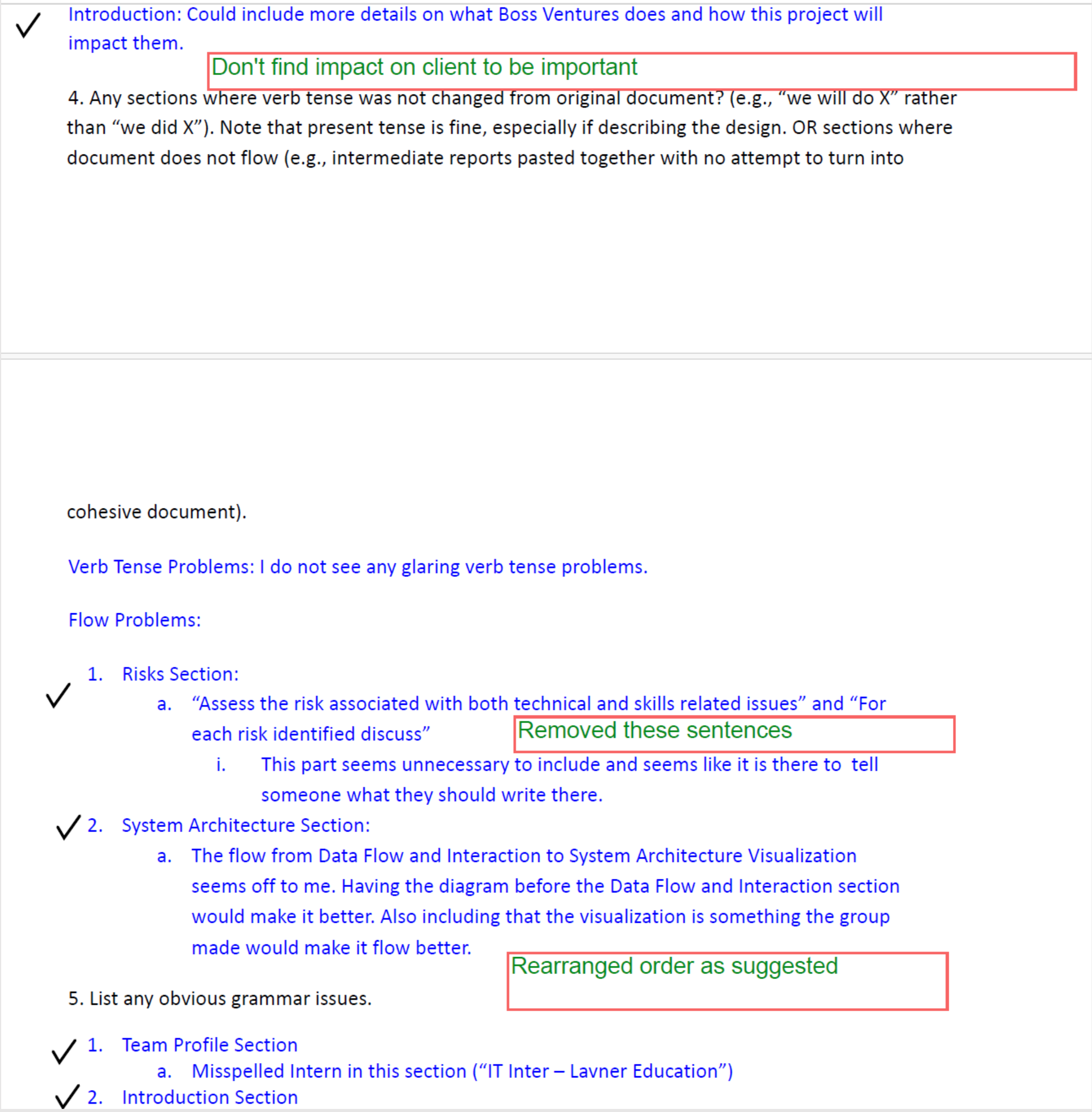
3. Where is more detail needed?

✓ Functional Requirements: Not clear on what “Mobile Optimizations” are. Could explain more in detail about this.

✓ Definition of Done: This section could have a lot more details instead of 2 bullet points. Explain more on the  definition of done.

✓ Project Ethical Considerations: Could explain more on what “Techno-Professional Ethical Conside

Don't feel a need to define as reading the details associated with them explains enough



✓ Introduction: Could include more details on what Boss Ventures does and how this project will impact them.

Don't find impact on client to be important

4. Any sections where verb tense was not changed from original document? (e.g., "we will do X" rather than "we did X"). Note that present tense is fine, especially if describing the design. OR sections where document does not flow (e.g., intermediate reports pasted together with no attempt to turn into cohesive document).

Verb Tense Problems: I do not see any glaring verb tense problems.

Flow Problems:

✓ 1. Risks Section:
   a. "Assess the risk associated with both technical and skills related issues" and "For each risk identified discuss"
      i. This part seems unnecessary to include and seems like it is there to tell someone what they should write there.

Removed these sentences

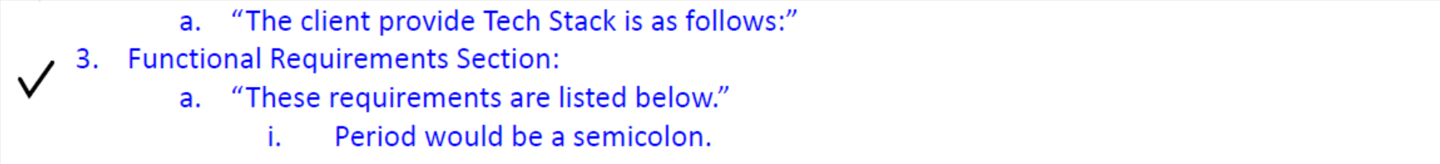
✓ 2. System Architecture Section:
   a. The flow from Data Flow and Interaction to System Architecture Visualization seems off to me. Having the diagram before the Data Flow and Interaction section would make it better. Also including that the visualization is something the group made would make it flow better.

Rearranged order as suggested

5. List any obvious grammar issues.

✓ 1. Team Profile Section
   a. Misspelled Intern in this section ("IT Inter – Lavner Education")

✓ 2. Introduction Section



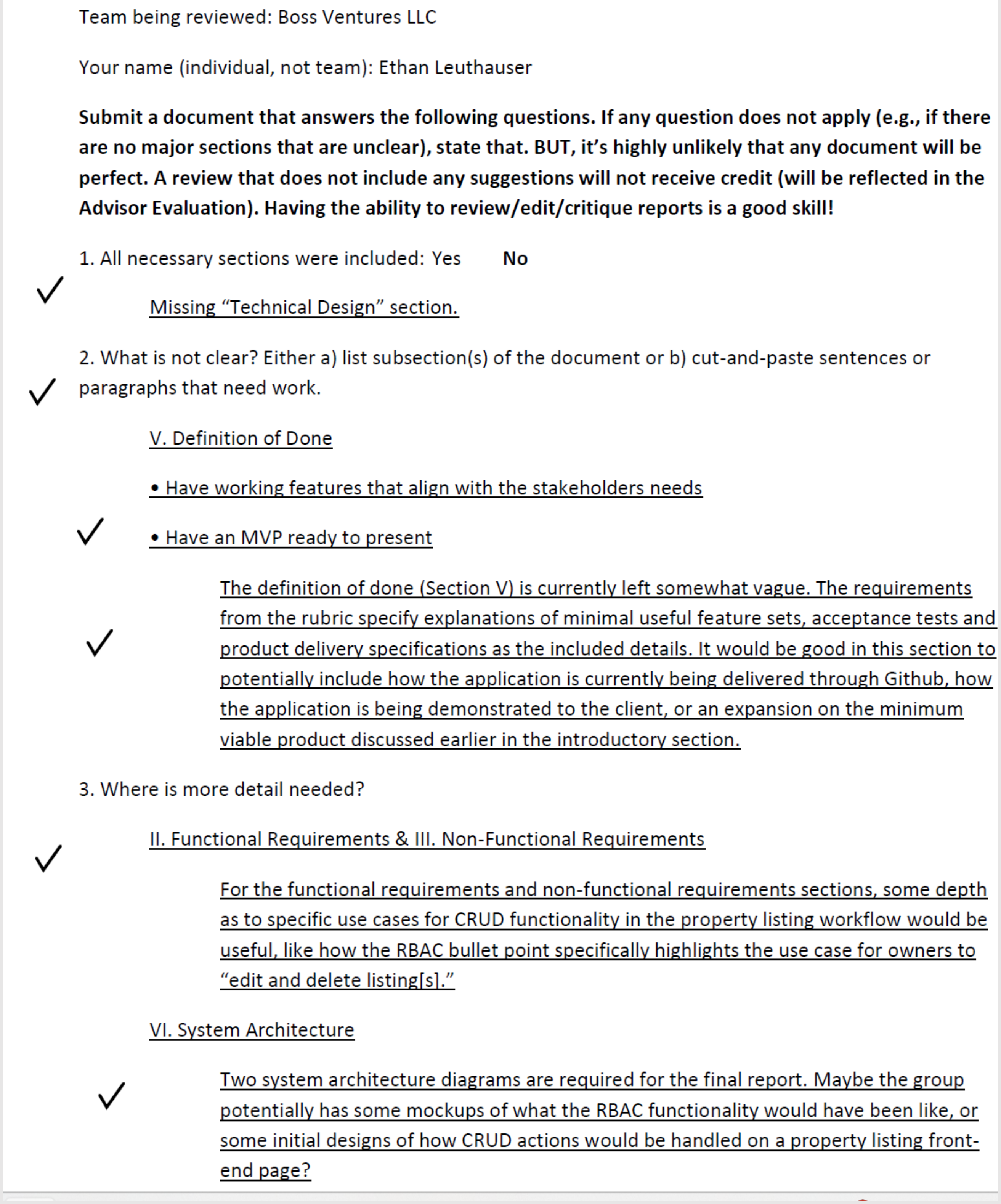
a. "The client provide Tech Stack is as follows:"

✓ 3. Functional Requirements Section:

a. "These requirements are listed below."

i. Period would be a semicolon.

## Review 2



Team being reviewed: Boss Ventures LLC

Your name (individual, not team): Ethan Leuthauser

**Submit a document that answers the following questions. If any question does not apply (e.g., if there are no major sections that are unclear), state that. BUT, it's highly unlikely that any document will be perfect. A review that does not include any suggestions will not receive credit (will be reflected in the Advisor Evaluation). Having the ability to review/edit/critique reports is a good skill!**

1. All necessary sections were included: Yes **No**

✓ Missing "Technical Design" section.

✓ 2. What is not clear? Either a) list subsection(s) of the document or b) cut-and-paste sentences or paragraphs that need work.

V. Definition of Done

• Have working features that align with the stakeholders needs

✓ • Have an MVP ready to present

✓ The definition of done (Section V) is currently left somewhat vague. The requirements from the rubric specify explanations of minimal useful feature sets, acceptance tests and product delivery specifications as the included details. It would be good in this section to potentially include how the application is currently being delivered through Github, how the application is being demonstrated to the client, or an expansion on the minimum viable product discussed earlier in the introductory section.

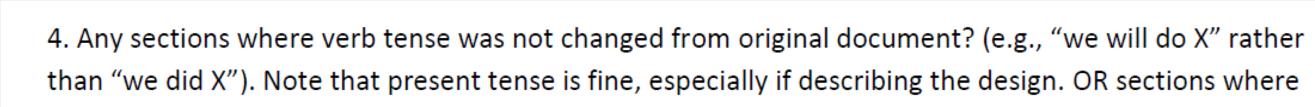
3. Where is more detail needed?

✓ II. Functional Requirements & III. Non-Functional Requirements

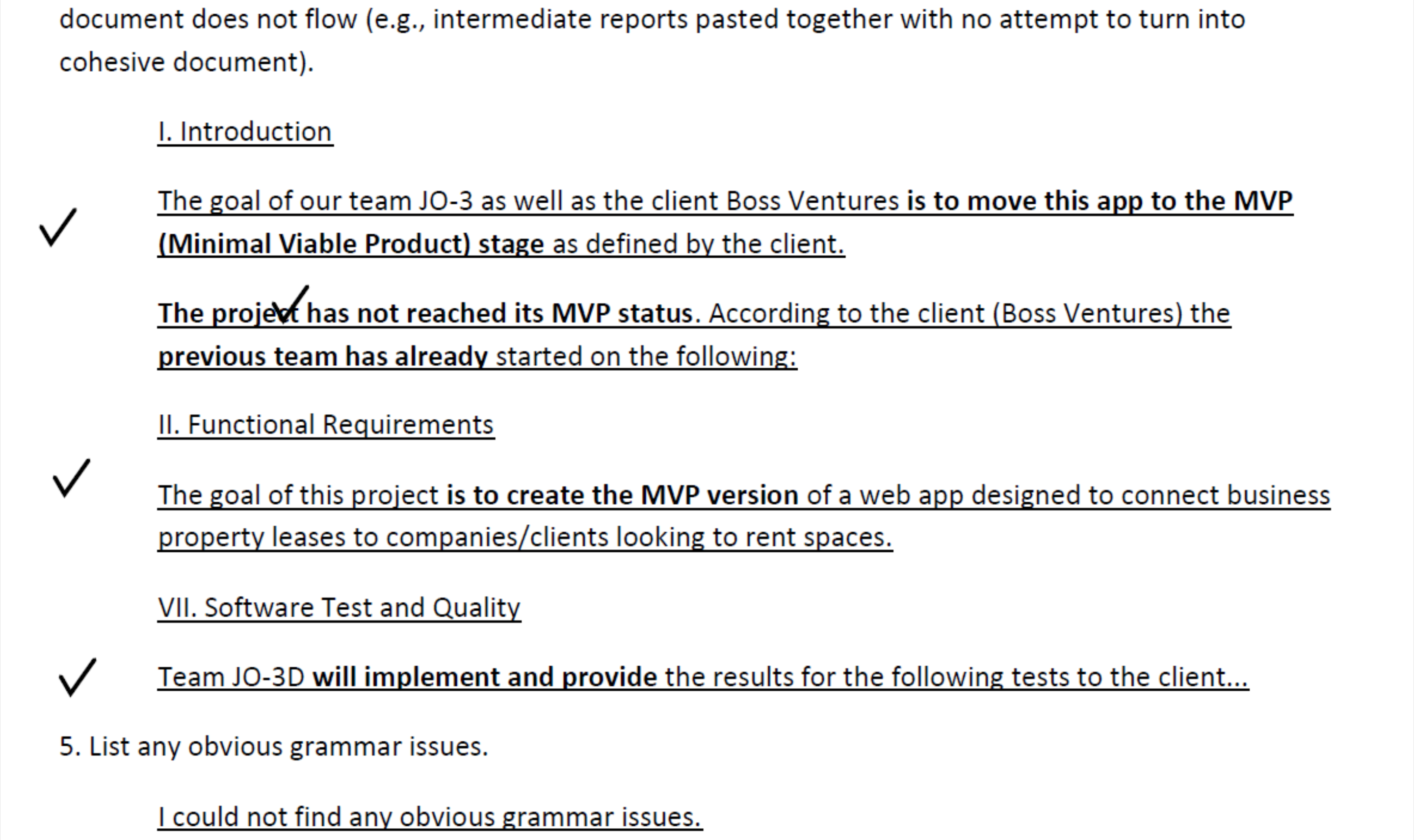
For the functional requirements and non-functional requirements sections, some depth as to specific use cases for CRUD functionality in the property listing workflow would be useful, like how the RBAC bullet point specifically highlights the use case for owners to "edit and delete listing[s]."

VI. System Architecture

✓ Two system architecture diagrams are required for the final report. Maybe the group potentially has some mockups of what the RBAC functionality would have been like, or some initial designs of how CRUD actions would be handled on a property listing front-end page?



4. Any sections where verb tense was not changed from original document? (e.g., “we will do X” rather than “we did X”). Note that present tense is fine, especially if describing the design. OR sections where



document does not flow (e.g., intermediate reports pasted together with no attempt to turn into cohesive document).

I. Introduction

✓ The goal of our team JO-3 as well as the client Boss Ventures **is to move this app to the MVP (Minimal Viable Product) stage** as defined by the client.

**The project ✓ has not reached its MVP status**. According to the client (Boss Ventures) the **previous team has already** started on the following:

II. Functional Requirements

✓ The goal of this project **is to create the MVP version** of a web app designed to connect business property leases to companies/clients looking to rent spaces.

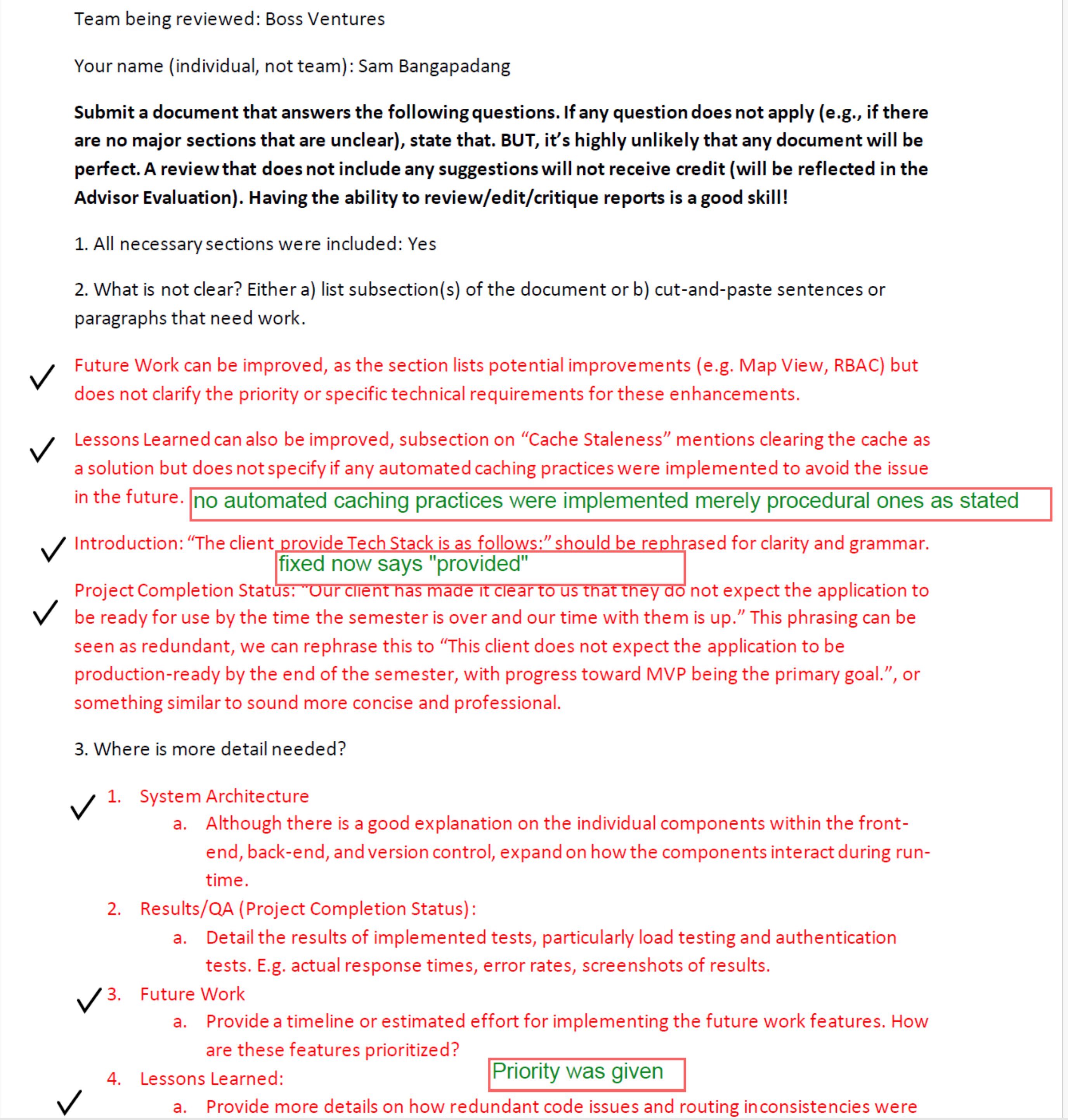
VII. Software Test and Quality

✓ Team JO-3D **will implement and provide** the results for the following tests to the client...

5. List any obvious grammar issues.

I could not find any obvious grammar issues.

## Review 3



Team being reviewed: Boss Ventures

Your name (individual, not team): Sam Bangapadang

**Submit a document that answers the following questions. If any question does not apply (e.g., if there are no major sections that are unclear), state that. BUT, it's highly unlikely that any document will be perfect. A review that does not include any suggestions will not receive credit (will be reflected in the Advisor Evaluation). Having the ability to review/edit/critique reports is a good skill!**

1. All necessary sections were included: Yes

2. What is not clear? Either a) list subsection(s) of the document or b) cut-and-paste sentences or paragraphs that need work.

✓ Future Work can be improved, as the section lists potential improvements (e.g. Map View, RBAC) but does not clarify the priority or specific technical requirements for these enhancements.

✓ Lessons Learned can also be improved, subsection on "Cache Staleness" mentions clearing the cache as a solution but does not specify if any automated caching practices were implemented to avoid the issue in the future.

no automated caching practices were implemented merely procedural ones as stated

✓ Introduction: "The client provide Tech Stack is as follows:" should be rephrased for clarity and grammar.

fixed now says "provided"

✓ Project Completion Status: "Our client has made it clear to us that they do not expect the application to be ready for use by the time the semester is over and our time with them is up." This phrasing can be seen as redundant, we can rephrase this to "This client does not expect the application to be production-ready by the end of the semester, with progress toward MVP being the primary goal.", or something similar to sound more concise and professional.

3. Where is more detail needed?

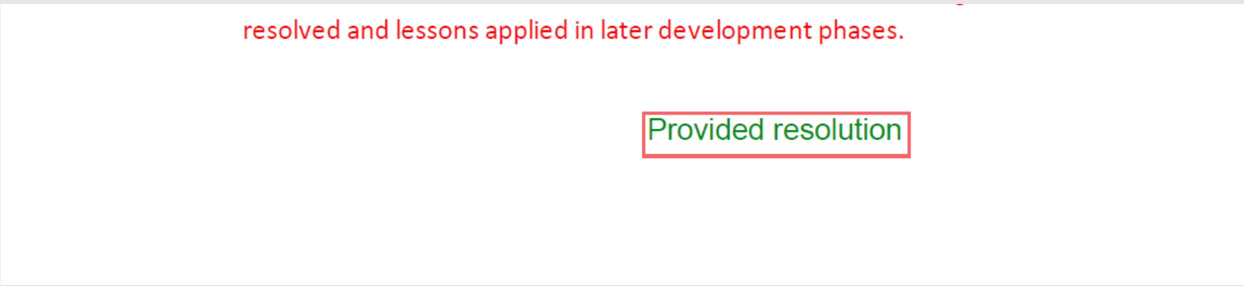
✓ 1. System Architecture
   a. Although there is a good explanation on the individual components within the front-end, back-end, and version control, expand on how the components interact during run-time.
2. Results/QA (Project Completion Status):
   a. Detail the results of implemented tests, particularly load testing and authentication tests. E.g. actual response times, error rates, screenshots of results.

✓ 3. Future Work
   a. Provide a timeline or estimated effort for implementing the future work features. How are these features prioritized?

Priority was given

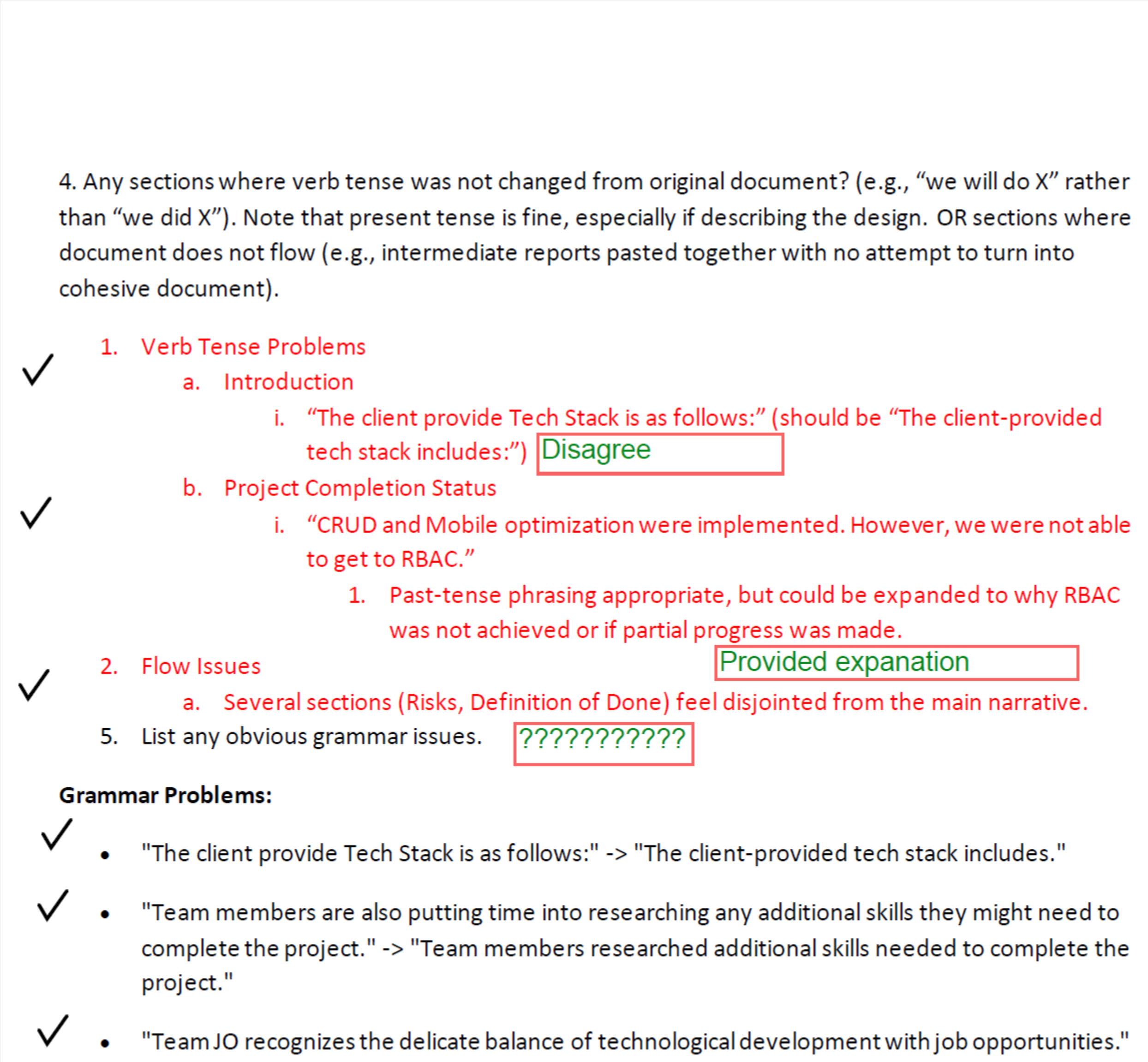
4. Lessons Learned:

✓ a. Provide more details on how redundant code issues and routing inconsistencies were



resolved and lessons applied in later development phases.

Provided resolution



4. Any sections where verb tense was not changed from original document? (e.g., “we will do X” rather than “we did X”). Note that present tense is fine, especially if describing the design. OR sections where document does not flow (e.g., intermediate reports pasted together with no attempt to turn into cohesive document).

✓

1. Verb Tense Problems
    a. Introduction
        i. “The client provide Tech Stack is as follows:” (should be “The client-provided tech stack includes:”) Disagree

✓

    b. Project Completion Status
        i. “CRUD and Mobile optimization were implemented. However, we were not able to get to RBAC.”
            1. Past-tense phrasing appropriate, but could be expanded to why RBAC was not achieved or if partial progress was made.

Provided expanation

✓

2. Flow Issues
    a. Several sections (Risks, Definition of Done) feel disjointed from the main narrative.

5. List any obvious grammar issues. ???????????

**Grammar Problems:**

✓

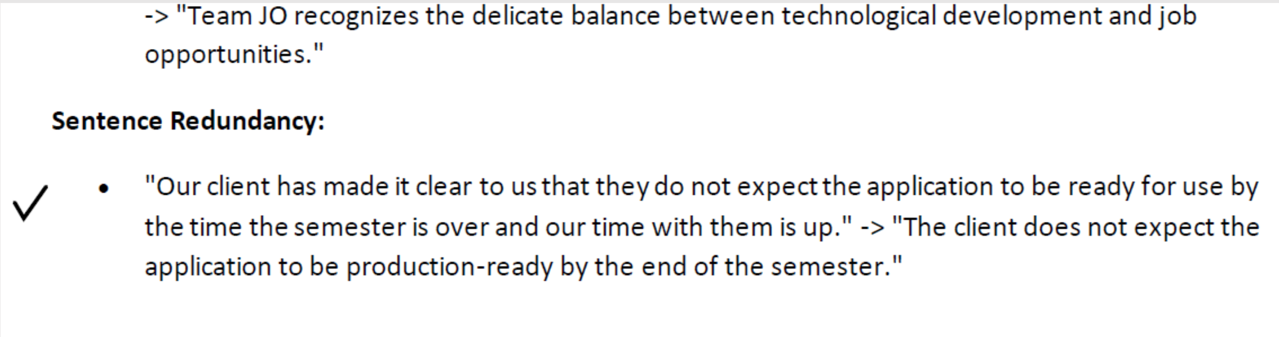
- "The client provide Tech Stack is as follows:" -> "The client-provided tech stack includes."

✓

- "Team members are also putting time into researching any additional skills they might need to complete the project." -> "Team members researched additional skills needed to complete the project."

✓

- "Team JO recognizes the delicate balance of technological development with job opportunities."



-> "Team JO recognizes the delicate balance between technological development and job opportunities."

**Sentence Redundancy:**

✓ 
- "Our client has made it clear to us that they do not expect the application to be ready for use by the time the semester is over and our time with them is up." -> "The client does not expect the application to be production-ready by the end of the semester."

# References

All references must be numbered in the order they are cited within the text
Formatted per IEEE format
Be sure to include citations for interviews.

[1] "ACM Code of Ethics and Professional Conduct," [Online]. Available: https://www.acm.org/code-of-ethics. [Accessed 19 October 2024].

[2] IEEE-CS/ACM joint task force on Software Engineering Ethics and Professional Practices (SEEPP), "Code of Ethics: IEEE-CS/ACM Joint Task Force on Software Engineering Ethics and Professional Practices," [Online]. Available: https://www.computer.org/education/code-of-ethics. [Accessed 19 October 2024].

## Appendix A – Key Terms

Include descriptions of technical terms, abbreviations and acronyms

| Term | Definition |
|---|---|
| *API* | **API** – and API or Application Programming Interface, is a set of rules that allow software applications to communicate with each other. APIs can make software development faster and easier by allowing developers to integrate data, services, and capabilities from other applications |
| *back-end* | The **back-end** is the infrastructure and data that allows an application to function. It's also known as the server-side and is made up of the server, application, and database. The back-end is responsible for storing and processing data for users, and for ensuring that everything runs smoothly behind the scenes. |
| *Firebase* | **Firebase** (backend / infrastructure) is Google's mobile and web app development platform that helps developers build apps and games used for authentication just a JavaScript function to add authentication |
| *Firestore* | **Firestore** is Google's NoSQL document database that's used to store data in documents organized into collections. It's designed to simplify the process of syncing, storing, and querying data for mobile, web, and IoT (Internet of Things) apps. Firestore is suitable for applications that require query ability, scalability, and high availability, such as those that involve live asset tracking, activity tracking, real-time analytics, and more |
| *Git* | **Git** – an open-source version control software, to enable real-time collaboration between multiple people. |
| *GitHub* | **GitHub** - a cloud-based platform that allows users to store, share, and collaborate on code and files. It's a web-based interface that uses Git, an open-source version control software, to enable real-time collaboration between multiple people. |
| *MVP* | An **MVP** is a Minimum Viable Product. As the name suggests, it requires the minimum amount of development to achieve a state where users may test it. By merely developing the minimum amount to achieve viability the cost of development is limited prior to testing. Therefore, if while testing the product users don't rate it well, costs associated with overdevelopment will be saved. |
| *Quasar* | **Quasar** is a component library of Vue (see Vue.js in here in Appendix A) |
| *Pinia* | **Pinia** is a store library for Vue, it allows you to share a state across components/pages and keep any existing state while developing |
| *serverless back-end* | A **serverless back-end** is a platform that eliminates the need for developers to deploy and maintain web servers and other common back-end features like authentication or database access management. |
| *Virtual DOM* | **Virtual DOM** (Virtual Document Object Model) is a programming concept where a virtual representation of a UI is kept in memory synced with "Real DOM" by a library such as ReactDOM and this process is called reconciliation. Virtual DOM makes the performance faster, not because the processing itself is done in less time. |
| *Vue.js* | **Vue.js** – an open-source JavaScript framework for building user interfaces (UIs). Vue.js is built on top of HTML, CSS, and JavaScript, and uses a component-based programming model to help developers efficiently create UIs. It is a Virtual DOM (see Virtual DOM here in the Appendix) |

| | |
|---|---|
| *escaped* | **Escaping** means treating special SQL characters like single quotes (`'`), semicolons (`;`), and others as literal text. This is used in reference to SQL injection attacks being turned benign by the use of “escaping” any potentially harmful characters rendering them literal text rather than destructive SQL command elements |