SOEN 390 - Capstone Software Engineering Design Project
Team 12 - Deliverable 1

ElevateNet System Architecture Design

Winter 2023

Student ID	Student Name
40173635	Jasmit Kalsi
40177866	Fatema Akther
40175616	Leon Zhang
40176360	William Chong
40175335	Julie Trinh
40060174	Zahin Khan
40158192	Flora Avakian
40107680	Lyes Kara
40114478	Sharjanan Staniculas

Contents

1 Introduction

1.1 Identifying information

ElevateNet is a job search platform, meaning it works better using a microservices architecture. This means that the platform is broken down into a collection of small, independent services that communicate with each other over a network. Each service is responsible for a specific function, such as managing job listings, handling search queries, or displaying candidate profiles. These services will be implemented on the application which will be hosted on a responsive website.

This architecture allows it to handle a large volume of data and traffic, and to evolve and adapt quickly to the users needs. Each service can be developed, deployed, and scaled independently, which allows for efficient and flexible development. Additionally, this architecture allows for easier debugging and maintenance as each service can be handled independently. Furthermore, hosting on a responsive website makes it accessible to a larger demographic of people since it will work with new versions of operating environments which have browser access and not be restrictive.

Microservices architecture enables ElevateNet to have a more efficient and scalable system, better performance and faster delivery of new features and functionalities. This is why this architecture is common in large-scale web applications, that needs to handle a large volume of data, traffic, and to adapt quickly to the user's needs.

1.2 Supplementary information

A change history will be found on Github in order to track and serve as version control during the construction of the system. Every commit and pull request will be registered and logged onto a history in order to trace all the changes.
Github will also serve the quality control team that will review all the pull requests before they can be pushed onto the main branch of the repository. The team will serve as an approving authority that will make sure that all the files pushed onto that branch are functional, well constructed and serve a purpose.
The system will be issued on April 22nd.

1.3 Other information

1.3.1 Overview (optional)

ElevateNet is a professional networking platform mainly looking to facilitate the process of job seeking and to build solid communities and professional connections in the work field.

ElevateNet is a networking platform that caters to the business needs of its users. It enables professionals to connect with one another, search for job opportunities and manage job

postings. It is compatible with all devices supporting web browsing including mobile phones. Professionals all over the world will be able to network on the platform.

We also included a domain diagram and a component to help visualize the structure of our system, its content and behavior.

Stakeholders have important roles in the direction the system will follow. The product owner dictates the requirements and the definition of the product vision and strategy that will be followed. Below we have a more detailed descriptions of them and their role in the project and system.

1.3.2 Architecture evaluations

Microservices architecture is a software design pattern that structures an application as a collection of small, independently deployable services. Each service runs a specific business function and communicates with other services through a well-defined API.

Advantages	Disadvantages
Scalability: Individual services can be scaled up or down as needed, rather than scaling the entire application.	**Increased complexity**: With more moving parts and services, the overall system can be more complex to understand and manage.
Flexibility: Services can be developed and deployed independently, allowing for faster development and deployment cycles.	**Network Latency**: Services may be deployed on different machines, leading to increased network latency and potential communication issues.
Resilience: If one service fails, the others can continue to function, improving overall system reliability.	**More difficult to tes**t: Testing a microservices architecture can be more difficult as each service must be tested independently.
Easier to understand and maintain: a monolithic architecture can become complex and hard to understand, with a microservices architecture the system is broken down into smaller, more manageable pieces.	

In conclusion, Microservices architecture can bring many benefits such as scalability, flexibility, and resilience. But it also comes with the cost of increased complexity, network latency and difficulty in testing. It's important to evaluate the trade-offs and choose the architecture that best fits the specific requirements of the system.

1.3.3 Rationale for key decisions

The rationale for decisions made in a microservices architecture typically stem from the specific requirements and constraints of the system being developed. As discussed previously, microservices architecture is scalable, flexible, resilient, and easy to understand and maintain, which are key factors we are looking for when choosing an architecture.

Additionally, it was decided that the application would be web based, and have the ability to be displayed both on web and mobile platforms in order to make ElevateNet more accessible at all times.

2 Stakeholders & Concerns

2.1 Users

1. **Product Owner**: As a product owner, they are responsible for representing the interests of all stakeholders and ensuring that the final product meets their needs and expectations. They are also responsible for defining the product vision and strategy, creating and prioritizing the product backlog, and communicating with the development team and other stakeholders to ensure that the product is delivered on time and within budget.

Concerns	Quality Attributes
Is the platform easy to navigate and use?	Usability
Is the data of the users well protected?	Security
Can the users use the platform as a tool to further their career elsewhere?	Interoperability

2. **Job Seekers**: As a job seeker, they would be looking to connect with professionals in their field of study, network with peers, and search for job opportunities. They may also be interested in learning about different industries and exploring career options. Additionally, they will use the site to showcase their skills and accomplishments, such as through a personal portfolio or resume.

Concerns	Quality Attributes
How simple is it to search for the job opportunities I'm looking for?	User-Friendly
Am I able to express myself to my viewers through my profile page?	Personalization

3. **Recruiter**: As a recruiter, their interests would align with finding new talent that they could recruit towards their company. This means having the option to find job seekers looking for employment, be able to view their profiles and also have an option to filter out job seekers that would fit their company the best. They may also attend job fairs and meet with college career centers to find potential candidates.

Concerns	Quality Attributes
Are events simple to find, and easy to set up?	User-Friendly
Am I able to effectively search for the candidates with the optimal skillset?	Relevance

4. **Company**: Using our platform to mark their presence among many others, companies aim to advertise their events, and products to users of all kinds in the pursuit of increasing their revenue. To do so, these corporations seek a platform that reliably distributes their postings, and steadily increases the reputation of their brand through frequent exposure.

Concerns	Quality Attributes
Will the platform convey our announcements to the right audience?	Relevance

4 Views+

4.1 Use Case View (Use Case Diagram)

[To be implemented on Sprint 2.]

4.2 Logical View (Domain Diagram)

The domain diagram for the product revolves around the User object and its nearby connections. Such connections go from user-specific actions, lists, and groups that relate to the functionalities of ElevateNet within the context of the problem domain.

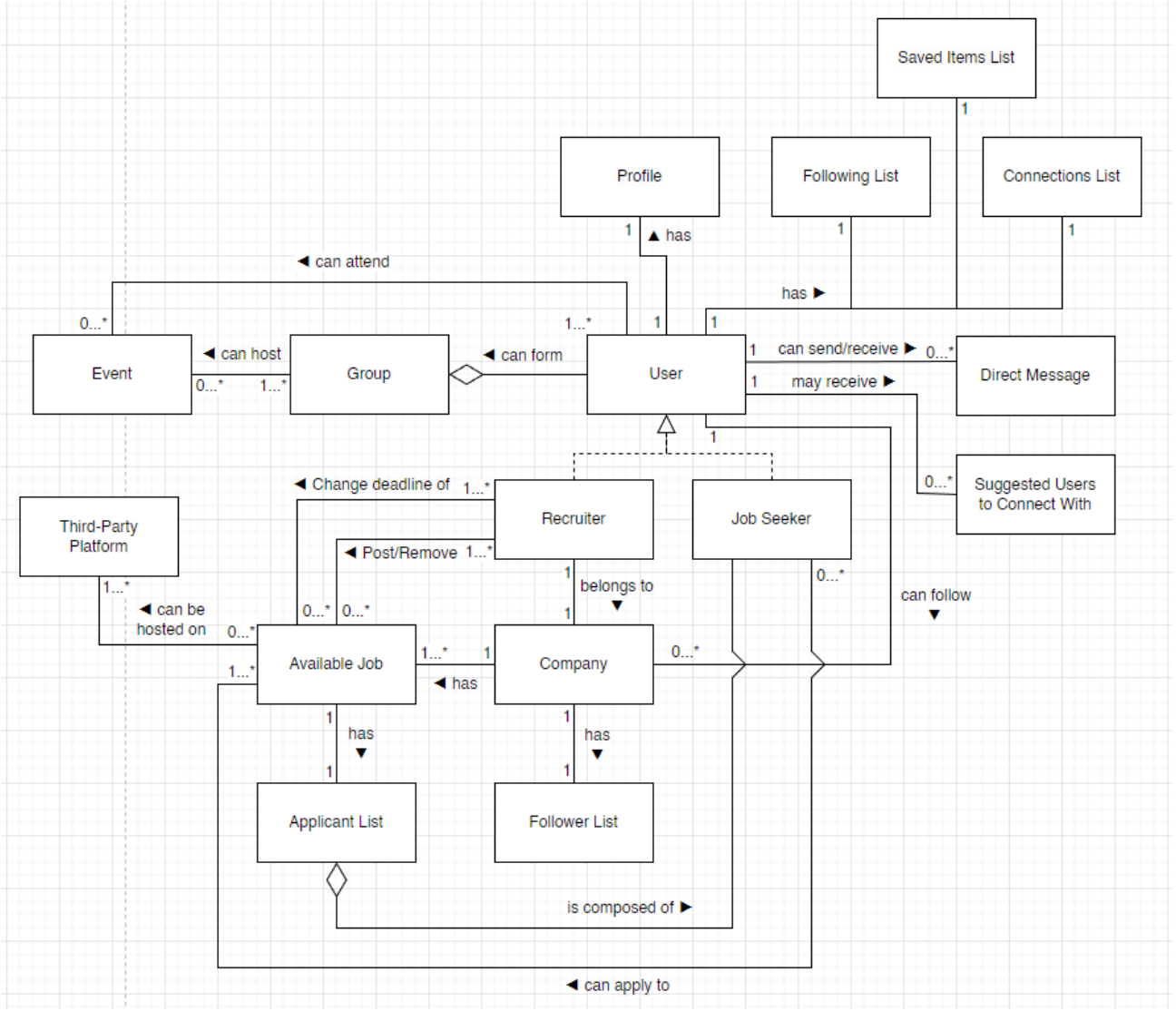

Figure 1. Domain Diagram.

The diagram seen in fig.1 revolves around the User object as the main conceptual class to which other objects serve to complement. Three main types of objects can be defined in order to simplify the diagram:

4.2.1 User

- **Job Seeker:** The main target demographic who lies in the domain problem. The users in this category are likely to use most of the features of the application.

- **Recruiter:** The other demographic that serves to maintain the flow of interest within the website. The users in this category are associated with a company and have privileges that allow them to operate on job applications.

4.2.2 Entity

- **Company:** Any job's associated owner that seeks the ideal candidate for any needed positions. This entity can be followed by Users for news, and has listings of jobs.

- **Group:** A collection of Users that can host events on the platform.

- **Third-Party Platform:** Any external platform that has permission to host the available jobs present on ElevateNet.

4.2.3 List

- **Following:** A collection of users/companies that a User follows.

- **Saved Items:** A collection of posts or other items that a User can safekeep for later use.

- **Connections:** A collection of users that share a common relation whether through work or similar following of a User.

- **Follower List:** A collection of users/companies that follows the User.

- **Applicant List:** A collection of job seekers who have applied for the specified position. Belongs to an Available Job object.

4.2.4 Miscellaneous

- **Direct Message:** An object that represents the messaging aspect of the platform for Users to use as a communication, and connection tool.

- **Suggested User to Connect With:** An object that serves to send recommendations to a User to expand their network, and connections.

- **Profile:** An object that can be personalized by a User as a means to intrigue recruiters, and other Users.

- **Event:** An object that is organized by a group of Users for promotion or other activities for a cause.

4.3 Process View (Sequence Diagram)

[To be implemented on Sprint 2.]

4.4 Implementation View (Component Diagram)

The implementation of our ElevateNet platform consists of two major components that relay information to one another through Google-backed application development service called Firebase. Using the in-built features of the aforementioned software, the blueprint of the system components relative to each other can be seen in the figure below.

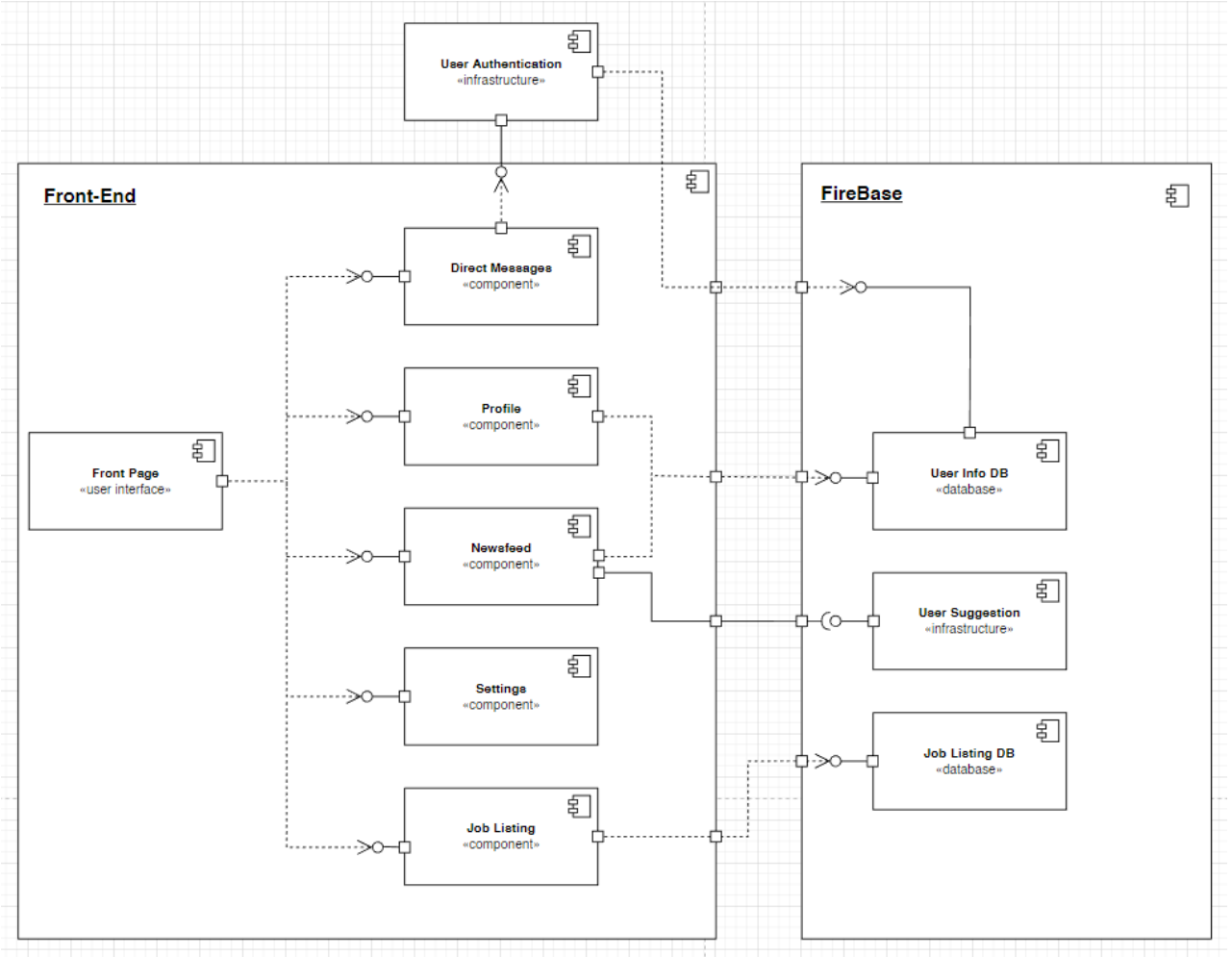

Figure 2. Component Diagram.

On the highest level of abstraction, it can be seen that the pattern followed is a basic front-end to back-end style of implementation, with Firebase offering the database management, and user authentication. While the front-end focuses on delivering direct messages, profile page rendering, newsfeed generation, settings manipulation, and job listing features, the back-end processes the dependencies and accesses to the crucial user info & job listing database. It should be noted that the news feed component may utilize the user suggestion interface to enhance the job searching experience by returning more relevant job listings.

4.5 Class View (Class Diagram)

[To be implemented on Sprint 2.]

4.6 Activity View (Activity Diagram)

[To be implemented on Sprint 2.]

4.7 Deployment View (Deployment Diagram)

[To be implemented on Sprint 2.]