Social Interest e-Club
Architecture Notebook

1. Purpose

Software architecture works as the main plan to show on a high level how the system works and how complex the system will be. Architecture Notebook is the document that describes and expands on the architectural overview of the system we are developing so architecture notebook collects architecturally how the system works in one document. These complexity and explanations include many ideas like; assumptions, decisions, constraints, justifications, architectural properties such as architectural goals and philosophy and architectural mechanisms. With this document tracing and following our goals during the development will be easier because we will have a blueprint to follow.

2. Architectural goals and philosophy

Architectural goals are the required properties and explanation of these properties. When these goals are written on a document this helps with preventing the risks that could occur during the development of the system. These requirements are the waypoints for the design decisions chosen.

Architectural philosophy is used to make sure that the system's development process is working correctly and to make sure that the system being developed has a long lifetime and these philosophies are helpful for robustness, scalability, security, flexibility and reusability. The system should be robust for long term maintenance.

The system will be a web application so the application as long as is supported by the browser doesn't have hardware dependencies. To prevent the performance issues the system should be able to function under unusual circumcisions such as too many users visiting the web application. The system's code should be clean and easy to understand because the system should be robust as stated above for long term maintenance. Also for the user experience to be very good the design of the web application should be clean, easily understandable and aesthetic to appeal to a larger population of users. For the same reason which is reaching a larger user base the application's language is going to be English.

3. Assumptions and dependencies

Our system will be a web application assuming that we will reach a bigger population of users and web application was chosen rather than mobile application because some of the members of the team are more familiar with web application development process but generally the team doesn't have in depth knowledge or experience about development yet so while development we have to learn how to develop a web application as well.

We are using javascript and java because we are much more experienced in writing code with Java rather than C# and also there are many tutorials on how to develop web applications with javascript and we chose VueJS on frontend development because it seemed to be easy to learn.

We are also using Figma for our design stages because this program is easy to learn and simplifies the design process.

4. Architecturally significant requirements

- System should be available 24/7.
- The system requires stable internet access.
- At the same time several users should be able to use the web application.
- The web application should be compatible with browsers.
- Loading time between pages of the application should be short.
- User experience should be easy to understand and access.

4.1. Why VueJS?

https://www.netguru.com/blog/why-vue-js#:~:text=a%20brief%20recap.-,Vue.,performance%20issues%20and%20scales%20well.

https://www.peerbits.com/blog/vuejs-is-growing-international-development-community.html

4.2. Why Spring?

https://spring.io/why-spring#:~:text=Spring%20makes%20programming%20Java%20quicker,world%27s%20most%20popular%20Java%20framework.

4.3. Java vs C#

https://www.educba.com/c-sharp-vs-java-performance/

4.3. Web application vs mobile application

https://sagaratechnology.medium.com/mobile-apps-vs-web-apps-which-is-the-better-option-868106c88730

https://careerfoundry.com/en/blog/web-development/what-is-the-difference-between-a-mobile-app-and-a-web-app/#:~:text=Web%20apps%20need%20an%20active,Web%20apps%20will%20update%20themselves.

5. Decisions, constraints, and justifications

- The programming languages we used are Java and JavaScript because they are object oriented languages and thus very helpful for future development of the system.

- For frontend development we are using Intellij Idea and VueJS because it is rather new, trendy and useful and easy to learn. And because in our team no one had any experience about software development so we decided to choose programs that are easier to understand and easier to find tutorials for.

- As for backend we are using Intellij Idea and postgresql and Postman. Postgresql is again rather easy to understand and open source. The schemas shown while the program is working is very useful. Postman is again very simple and clean and we could find our errors and test our inputs effortlessly.

- We chose to develop a web application and not a mobile application because we hadn't had any development experience and we could find a lot of recourses for website development and also with a web application we theorized that we would reach a larger population of users.

- The system's architectural design is Model View Controller model that way development gets simpler. MVC model is also discussed during BBM382 classes so we were more familiar with this model.

- Because of COVID-19 pandemic, as a team we are unable to meet up in person, this is one of the constraints we are facing.

6. Architectural Mechanisms

Duration

As long as user is logged in the session won't end.

Reliability

If a crash or error occurs the data like user's data, club' data etc. should be protected and saved.

Availability

The system will be open always so anytime a user wants to access the web application thay can use a web browser.

Scalability

As the circumcises change the system should be able to adapt to them so that errors and crashes don't occur.

Maintability

The amount of clubs and questionnaire is maintained by admin and specifically sub-club's responsibility falls on sub-club admins.

Security

The passwords of users will be hashed and will be irreversible.

Update frequency

When a new club opens the questionnaire should be updated.

User Interface

User interface will be clean and easy to understand.

Data Management

Data will be constantly updated according to changes that occur. For example if a new member registers than the dataset about members updates.

7. Key abstractions

Stakeholders and Users:

- Developers: The team that is developing the system.
- Non-registered members: Visitors that have not signed in to the web application, they can only search the clubs; they do not have any other privileges.
- Registered members: Users of the web application who are signed in an attend the clubs and sub-clubs.
- Sub-club admins: Assigned by the admin to their sub-clubs; for the members who made a request for the sub-club to open; responsible for the sub-club they are attending.
- Admin: They choose sub-club admins and attend to members, clubs and questionnaire.

Interfaces and Design:

- Admin Panel: Where admin can reach their responsibilities for example; add club, delete club etc.
- Club Page: Where the club feed and events of the sub-clubs will be shown.
- Search Page: This age is for non-registered members only for the purpose of looking up the clubs and sub-clubs.
- Login Page: The page where users log into the system.
- Sign in Page: The page where non-registered members can register into the web application.
- Member Profile Page: The registered members can access and change their status here.
- Questionnaire: Where the suggested clubs are decided.

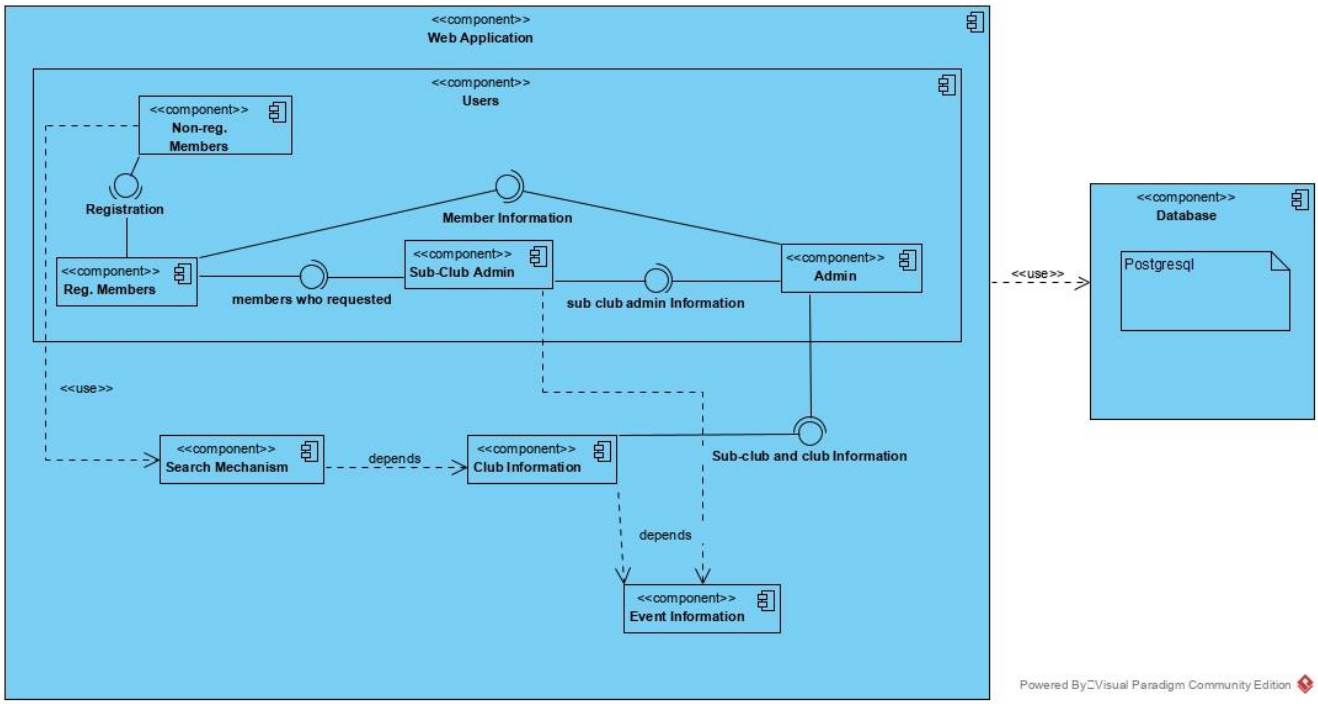
Models:

- Clubs
- Sub-Clubs
- Users shown above

Databases:

- Data collected and used; Postgresql is used for this purpose.

Component Diagram:



8. Layers or architectural framework

We are using spring for our development and spring framework consists of three architectural layers; presentation layer, business layer and database layer. In the presentation layer; the visualization is the main purpose, we are only developing a web application so we have one interface which consists of multiple pages and panels.

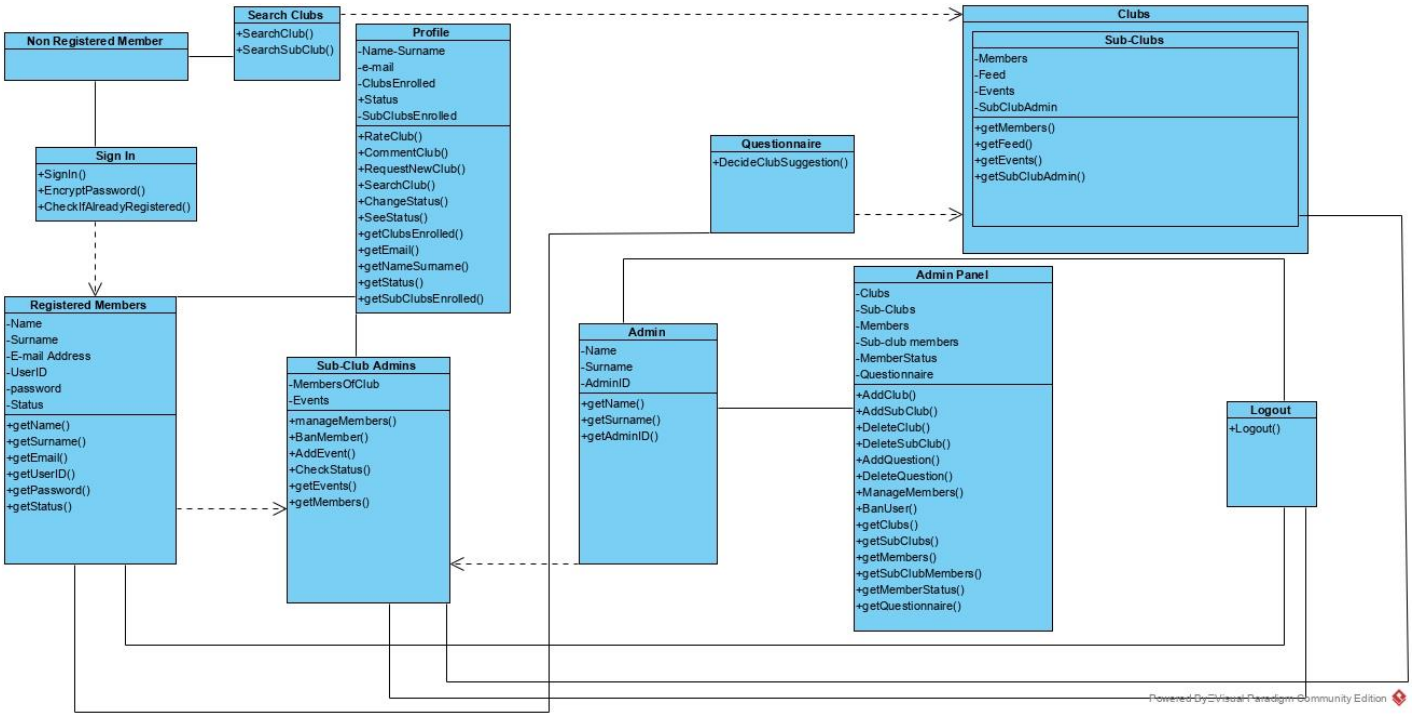
The second layer is business layer which contains the responsibility of being the connection between presentation and database layer. This layer acts according to the users' requests from and to system.

The third layer is database layer which is responsible with providing every necessary data and resources for the system. All of the layers are connected with each other.

9. Architectural views

4+1 Architectural View Model is used.

Logical View:

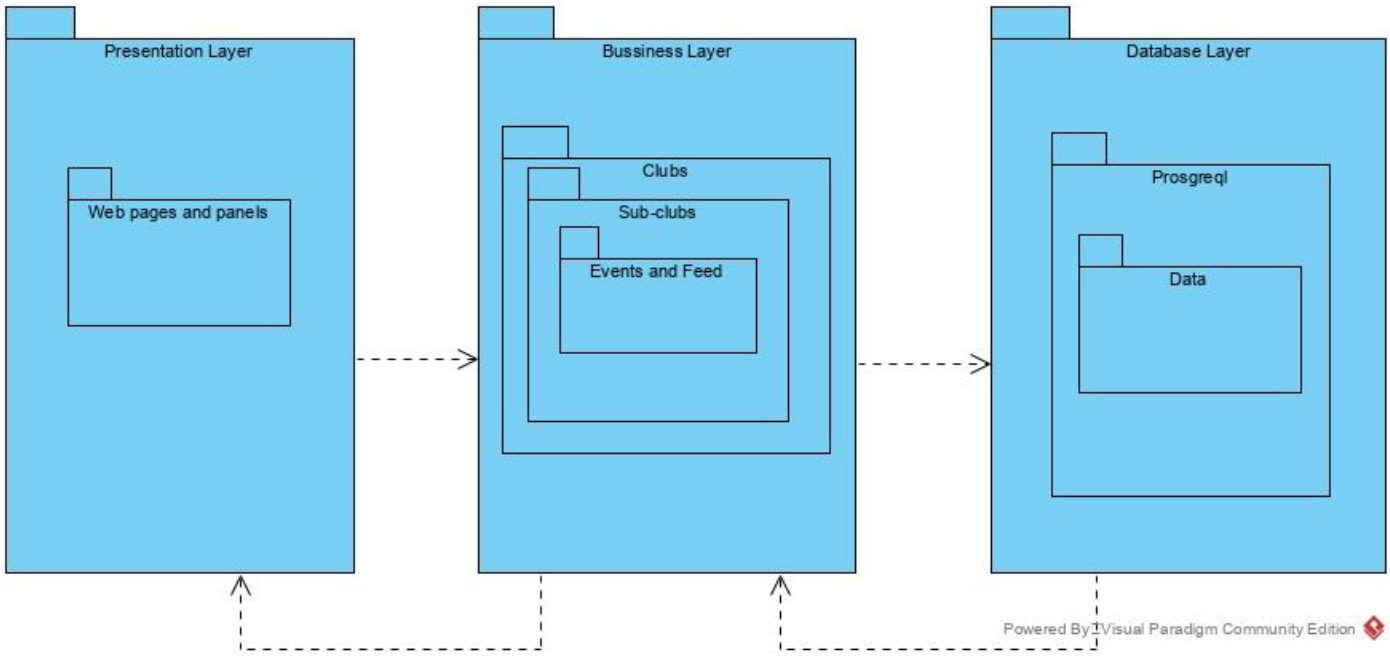


Class Diagram

Process View:

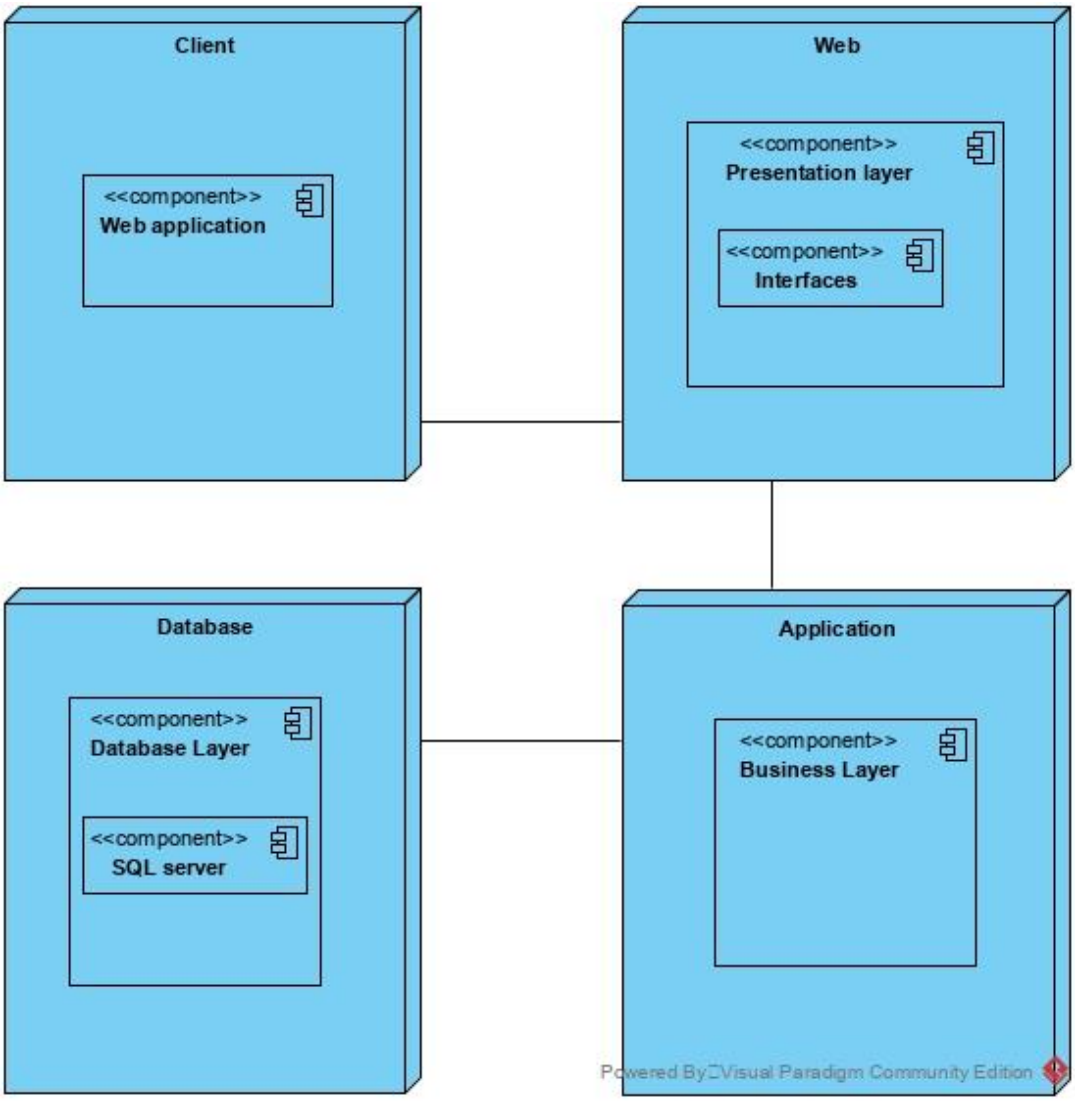
Activity diagrams were sent as Del2 - SSR attached file.

Development View:



Package Diagram

Physical View:



Deployment Diagram

Use Case Diagram: