Pingutopia - An Interactive City Builder

Created by Calvin Stanford

1. System overview

1.1 Problem statement

The ice caps are melting in the poles which is harming life as we know it. This is due to the warm water moving to the north and melting the ice Smith, G. (2000). With this happening it is resulting in animals living in these areas having their environments destroyed which can result in them going extinct. If this continues in the future then there will be no more north and south pole and the melted ice will result in the rise in water levels and affect livelihoods because there will be less land for us humans to live on. With the growing population and issues with us running out of land, we need all the land we can get.

1.2 System Description

A mobile client-server application that is an interactive city builder revolved around adapting to the environments based on real life events affecting the Antarctic. This is to bring awareness to the dangers of the poles. This is done through the environments that the penguins experience. Penguins require the land for them to survive. This is because when breeding that need to be on the land and also there primary predators that are in the sea such as leopard seals, fur seals, killer whales and sharks. So in this game you will be adapting your penguin inhabitants to the environments around you to ensure that you survive the elements.

This is where the user is required to assist the penguins in making decisions to help them grow a grand empire against the threats around them. This will require the users to allocate the penguins to specific roles based on their skills and traits that they have developed as they have grown up. This empire will be grown through methods of gathering, construction and helping young penguins grow up to help in the cause. The application is going to be developed by a software development team with the support of the other teams. This includes the management team, marketing team and the live services team. The management team will be there to make sure that everything is running smoothly and the teams are working to industry standards together.

The marketing team will be focused on interacting with the customers to understand what the needs for the customers are and provide this information to the teams. The live services team is to provide live information on the Antarctic that will directly affect the application in real time. This team will be working closely with the development team to ensure success between the application and the live data.

1.3 Purpose of this architecture

The architecture is developed to create ease of use for the software developers creating the application. This is provided by having a clear understanding of the features and requirements that are involved in the product. These requirements let the developers better understand what the client is wanting and create the product to more closely match what is expected with the quality attributes kept in mind.(Anubha Sharma, 2015) It is also to keep the complexity away from the user, which will require the user interface to be simple and reliable while having the functions complex.

1.31 Process for creating the architecture

Attribute Driven Design (ADD) will be the process I will be following when designing the architecture of PinguTopia. This process is focused on the quality attribute requirements; this will be done by obtaining the system's elements and implementing them into the tactics involved in the architecture. The requirements will be the driving factor on how the product will be produced. This is done by following ADDs cycle of “Plan, Do and Check” (Wojcik, R. 2006).

Plan is the quality attributes and the constraints that will need to be considered when designing the architecture.

Do is the requirements that will be needed for the quality attributes alongside functional requirements.

Check will be analysing the requirements to see if they have been met based on the design.

Functional and non functional requirements will be generated for the stakeholders and quality attributes to make sure that the system is providing exactly what is needed for success. The constraints will be requirements that the system will need to meet in the design. These constraints can be split into business and technological constraints.

The design will be documented through the 4 + 1 model (Philippe Kruchten, 1995).

1.32 Organisational context

PinguTopia will be developed by a small to medium sized software development company that will be designed from the mobile market as this is a space that has a lot of traffic currently (Kevin Curran, 2012). The typical user will be a child or young teen these are the primary users that we want to cater to due to them being the next generation that can make an impact on this issue.

The application should be easy to learn and understand how to use and have some gameplay elements that attract them to play longer, this will be done through the live updates that can impact your world. This will make them want to check up on their penguins to see if they are safe or managing well in the current environments. Due to it being a smaller project this is why the company size does not need to be large because there are many tools out there that enhance mobile development in the current market. But keeping the game fresh for the young user may be an issue for a small team which is where they could develop into a medium sized team later on.

1.4 Scope

The scope includes:

- The customers download and register to the application.
- The customers to develop their empire the way they want based on what is provided.
- The customers to view the live feed impacting their empire.
- The team monitoring the activity on the application servers.
- Notifications to customers on impacts made to their empire.

2. Architectural Drivers

2.1 Primary Functional Requirements

- The system should allow users to command the penguins to do various activities based on role(Hunter, Builder, Nurturer etc).
- The system must be connected to the live news feed to provide up to date information at all times through notifications.
- The system should allow users to display and modify their empire through edit mode.
- The system should translate process flow into executable code.

2.2 Technical Constraints

- Ensuring that the application is connected to the live news feed 99% of the time.
- The system must be able to store thousands of users' information at once and be secure.
- The system must be online 99% of the time.
- The system must be effective at filtering news feeds that are valid to avoid overload.
- The system needs to be stable while undertaking multiple online processes at once.
- The system must be able to withstand the flow of information from the live service information alongside the users.
- The development team must have a large understanding in the architecture to create a high level product.
- Security protecting users private data, so other users don't steal them

2.3 Business Constraints

- The business must provide the funding to keep the live service up 99% of the time.
- The team should commit at least 6 months to develop this product.
- The team should work on updates after the product has been developed.
- The business must get permission from companies that are providing live news feed information.
- The business must keep users' private data secure.

2.4 Stakeholder and primary concerns

Stakeholder	Concerns	Driver	View
Customer	Want more features and content	Have more to do on the app.	Functional Suitability - Functional Completeness
Development Team	Easy to maintain	When there are changes for the app it makes it easier to adapt them.	Maintainability - Modifiability, Testability
Marketing team	User Friendly	Easier for the team to advertise the app if it is easy to use for the customers.	Usability - Learnability, Accessibility, user interface Aesthetics
Management Team	Lower costs	So the app is more profitable.	Performance Efficiency - Resource Utilization
Live Service Providers	More Security	Due to the service being live it is more vulnerable to security breaches.	Security - Confidentiality, Accountability, Integrity

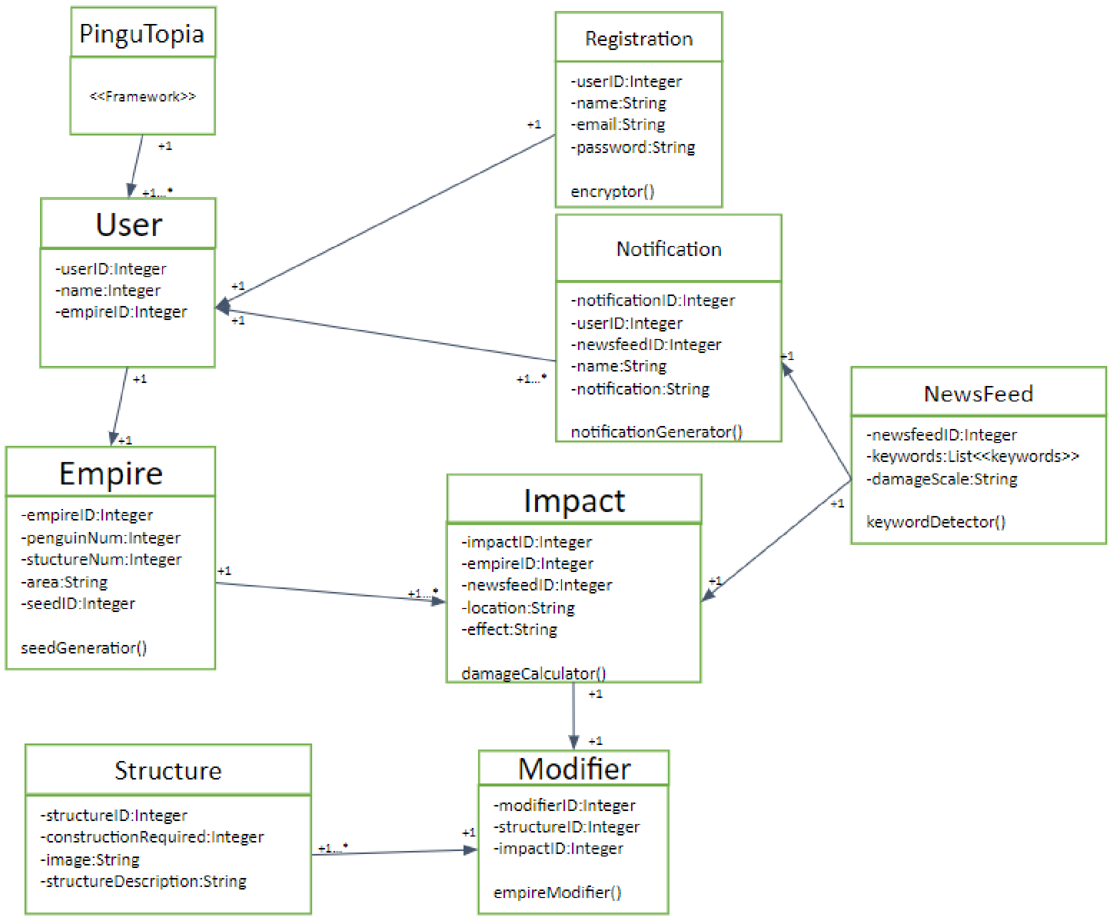
2.5 Quality attribute scenarios

Stimulus	Response	Time Response (man days)	It concluded when...
Customers want a new feature added to the app.	Development team contacts and information on what the customer wants and brainstorms potential options for this feature and see if it provides a benefit to the application.	2 week	The development team has decided to add the feature to the application and make a plan to implement it.
A user creates an animation for the app for fun that customers like.	The team gets in contact with the user and asks for permission to use what they have created to implement into the application.	1 week	The team has developed the animation into the app and the user has joined the team to create more animations like the original to improve the applications quality.
A user wants to join the development team and has some great ideas for the application.	The team will review the user's background and experience alongside seeing if there are any weaknesses in the team they the user can fill in.	4 days	The team decides to add him to the team. The user should be added because a user of a product that is also working on the product has a drive to make the application great. With his ideas get the user involved in the discussions.

The marketing team has an idea on a current feature that could be changed to improve the usability.	The development team will review the current version and brainstorm possible solutions to improve the usability.	3 week	The feature is updated to be more user friendly.

3. Views

3.1.1 Logical View Packet 1



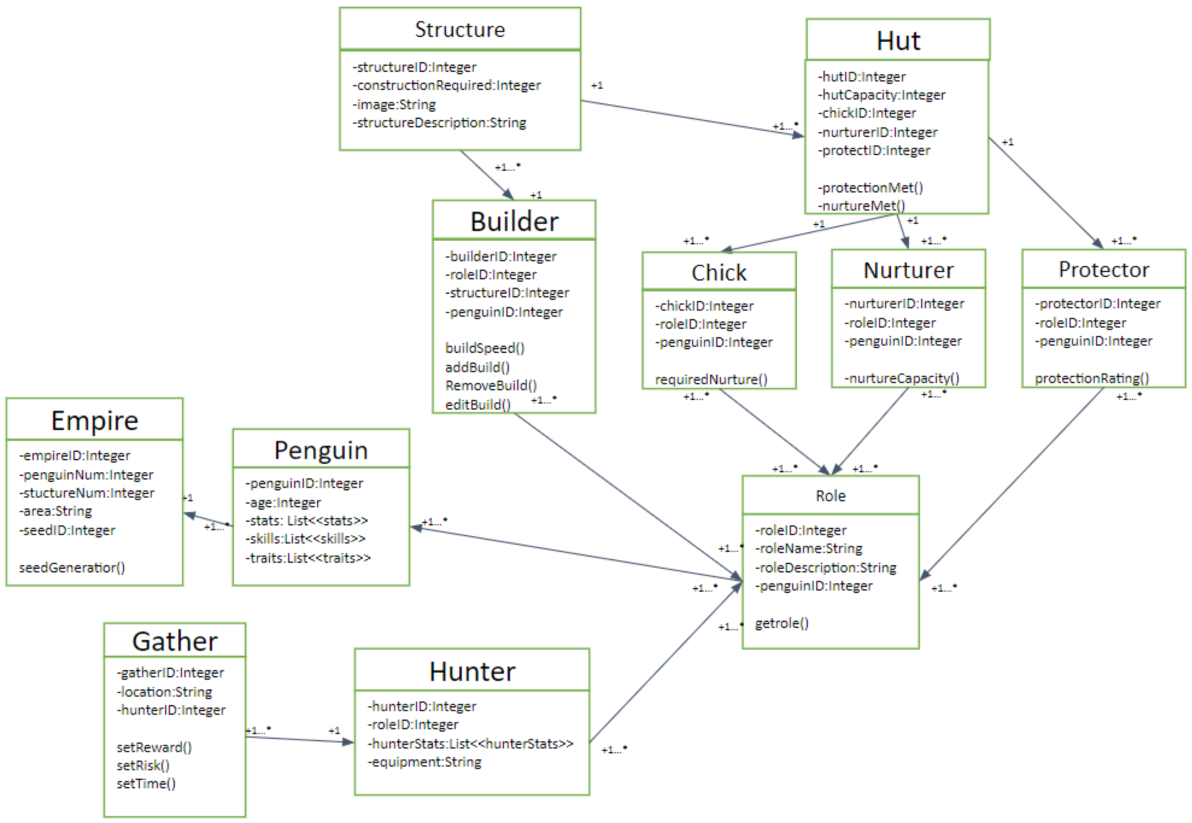
3.1.1.1 Element Catalogue:

Diagram 1

- **Pingutopia** is the whole framework connected by many classes and subclasses.
- **User** is first class connected to the framework as all the actions made by the user will be made or impacted by one of the other classes. This will store the user's name; it is the only information that needs to be public to give them some identity.

- **Registration** is where the user can make their Empire secure in the cloud. With the registration it will store private information that the user has input for logins if the user would like to access their Empire in various locations. This login is secure in the cloud due to the encryption process that will be built into the registration.
- **Empire** is what the user is building up in this application. This will be storing key information that will identify the user's empire from anothers. The seed will be the main way that the user's Empire layout will vary from another. This is because of the seed generator process that randomly generates an area layout for the user's Empire to begin development. The Empire class will also store and display structures and penguin numbers to see progression for the user.
- **Impact** is where the information is collected from Empire and Newsfeed class to calculate how the Empire is to be affected. This is done by getting the effect and location data from Newsfeed and using the damage calculator to decipher how this affects the Empire.
- **Modifier** gets the information from Impact and uses it to affect the Empire. This is done by gathering knowledge from the Structure class and making changes based on these aspects which is calculated from the empire modifier process.
- **Structure** is where all the information about different buildings that can be developed in PinguTopia and how they affect the Empire alongside how long it can take to develop one of these structures. It also stores the art that will be displayed on the application.
- **NewsFeed** is where information is gathered from the internet based on articles that are mentioning effects that are being made to the Antarctic. These articles will be determined based on websites that we deem credible and reliable and to extract information from these articles using the keyword detector process we will develop and these will be sent to impact calculations and to notifications to construct a sentence of relevance to the user.
- **Notifications** is where the information from news feed will be used to construct a sentence that represents what will happen to the users Empire and sent it to the users device as a notification so they are aware of the potential damage and check up on the Empire to see if the penguins are ok.

3.1.2 Logical View Packet 2



3.1.2.1 Element Catalogue:

Diagram 2

- **Penguins** is where all the information about each individual penguin is stored and displayed to the user. The stats will be built up based on the penguins activities. For example if the penguin is involved in hunting then the agility and strength of the penguin will develop. The skills of the penguin will also come from the penguins activities but will randomly come about as perks to benefit the empire, for example a nurturer may develop a skill that improves the performance of other nurturers around them. This skill would be called mentor. Traits will be randomly generated effects which are developed as the chicks grow up. These are similar to skills but can have both positive and negative effects and the empire may need to adapt around some of these penguins traits. For example a penguin may end up with the individual trait, this makes the penguin perform better in environments where they are alone.

- **Role** is where each of the individual penguins can be assigned a role to help with the development of the empire, this class stores and displays what roles are available and a description of what the penguin is expected to do when assigned this role. In the current architecture there are 5 roles for the penguins, builder, hunter, chick, nurturer and protector.
- **Builder** is the role where the penguin is expected to construct buildings to help with developing the empire. This is done by collecting information from the structure class and giving the user options of structure based on the penguins stats. All structures have a requirement from the penguin to meet before having availability to build. The speed that it is built is also calculated based on the penguins stats. Intelligence and strength will be what the penguin requires for the penguin to thrive as a builder. The builder will also have the option to remove and edit the building, maybe moving the building somewhere else so it is more protected could be a good idea.
- **Hunter** is the role where the penguin is expected to go out and collect food for the empire because without food the empire will starve and maybe even die if not dealt with. The hunters can be equipped with equipment such as spears to assist in the adventures they experience.
- **Gather** is what the hunters are set out to do. This is where the user selects a location for a hunter to explore and gather. These locations can have various strengths and weaknesses that follow them. Which are calculated from the risks, rewards and time taken. Risks could involve chance of success or even chance of death. Rewards could include finding new penguins to join the empire or even new equipment alongside potential bonus food to be found. The time taken will be based on hunters stats and the location selection.
- **Chick** is a role set automatically to young penguins that have not grown up enough to help in other roles. These chicks will be kept safe in huts until grown up enough to help with the growth of the empire. Each chick has their set required nurturing to have effective growth.
- **Nurturer** is a role that helps with the growth of the chicks they are key to growing chicks that will be a great help to the empire. Without nurturing the chicks could have many negative traits due to the lack of nurturing. Each nurturer has their set number of chicks they can take care of based on their skills and stats.

- **Protector** is a role that focuses on protecting the nurturers and the chicks that are in the huts as there can be predators that may come and attack the empire. Their protection effectiveness is based on protection rating which is calculated based on the stats of the penguin.
- **Hut** is a structure that can be built by a builder which is very important for the empire's growth. This is because this is where chicks are safely nurtured by nurturers. These huts if they want to thrive will need protectors in the event of an attack from predators such as sea leopards. There are many calculations that figure out how safe the hut is, how many chicks can stay there and whether the nurturing is sufficient for the chicks.

3.1.3 Variability Guide

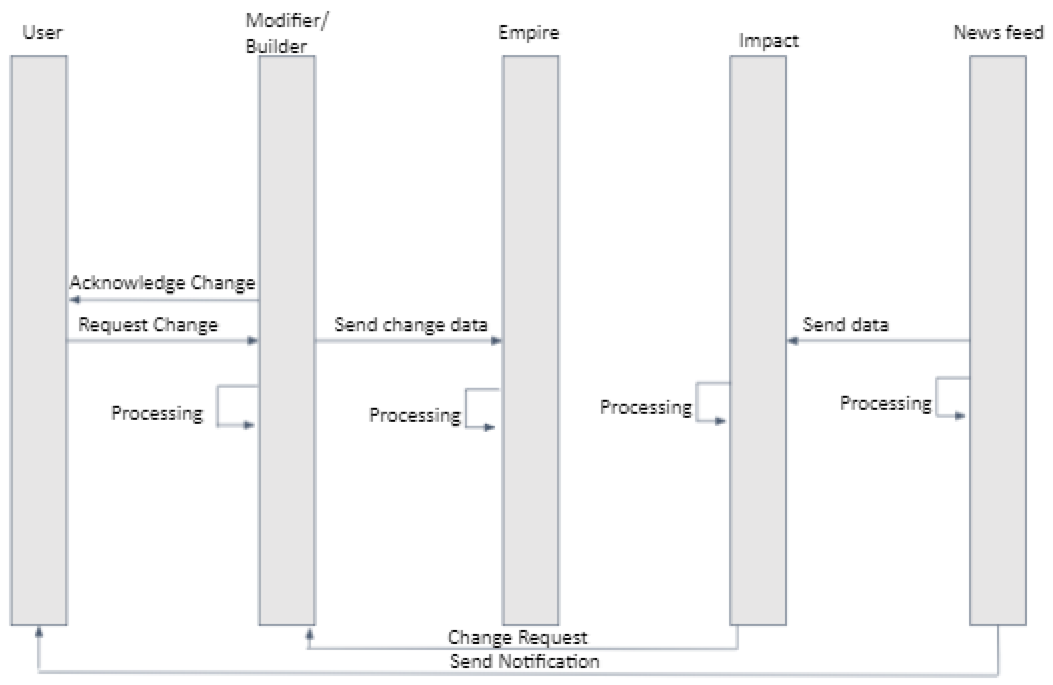
This architecture is flexible and adaptable as this is the base of the application and has many elements that can be expanded on such as the roles, there can be more roles and responsibilities added for the user to manage. There can be a goal to the game, or even an ending e.g. saving the antarctic. The news feed system could be adapted to have a more effective method to extract information or even from sources outside of articles. There could be a local storage system implemented so the user does not have to register and risk private information being leaked.

Other options is to have the location be in the arctic so instead of penguins it could be polar bears. One addition that would be nice is real time weather such as the one represented in flight simulator and there could be events that can correlate to the weather. To allow a larger range of users there could be language support. The monetization can be implemented in many different ways. One can be through advertising for example allowing users to watch advertisements for bonus rewards or there could be items that can be purchased for their penguins. These are all things that can be considered at a developmental level.

3.1.4 Architecture Background

The class diagram was selected to create a detailed overview of the structure that provides a solid architecture while giving availability for adoption. To make the structure easier to understand I broke it down into its two main systems: penguins roles and empire impact. This makes understanding them individually easier so that when you look at them as a whole it becomes clear what it expected. The reason for a class diagram is to display the objects that make up the system and how they relate with each other. That is exactly what I wanted to accomplish with my logical view. It is a great way of presenting both systems and subsystems when explaining a product which is key to understanding how all the systems come together to make a successful product. A class diagram was chosen due to its simple design so it makes it easy to edit while being easy to read while providing large amounts of detail and creating a story structure with a flow to it so you can see how everything comes together. It is important when creating a class diagram to provide clear names for classes and variables so that it can make sense to anyone reading it. Also allowing flexibility is important but not too much as this can create confusion on what is expected.

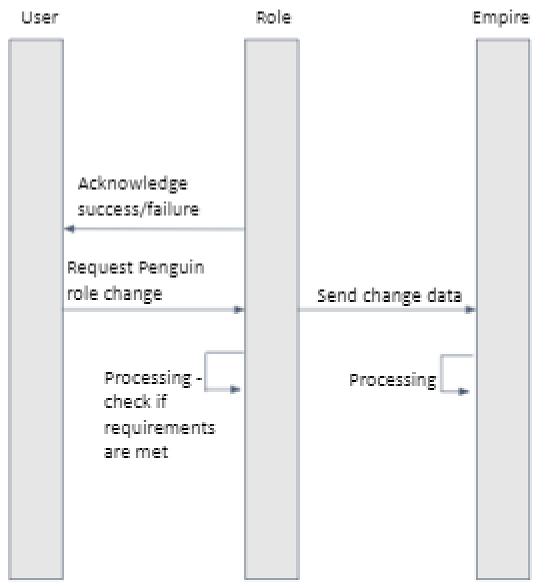
3.2.1 Process View Packet 1



3.2.1.1 Element Categories

- **Users** can send a process that requests change to the Empire based on the availability he has from the builder penguin they have selected. When the Builder receives this request it processes it so that it is sent as data to the Empire to be processed and updates the visuals based on the changes then stored in the database.
- **Newsfeed** when it finds an article it processes the keywords collected and sends them to both the User as a message and Impact to process into a change to send to the Modifier. The Modifier then processes the change request to convert it into language used by the application to send to Empire so that the changes can be made visually then stored to the database.

3.2.2 Process View Packet 2


User
Role
Empire
Acknowledge success/failure
Request Penguin role change
Send change data
Processing - check if requirements are met
Processing

3.2.2.1 Element Categories

- **Users** can select a penguin and send a request to change that penguins role. When this is done the Role class processes this request to see if the penguin is viable to be that role, if not then the request will be declined and the User will be notified of this. If the request is successful then the User will be notified of this success. The successful request is then sent to the Empire class to process and change the penguins class and store that data in the database.

3.2.2 Variability Guide

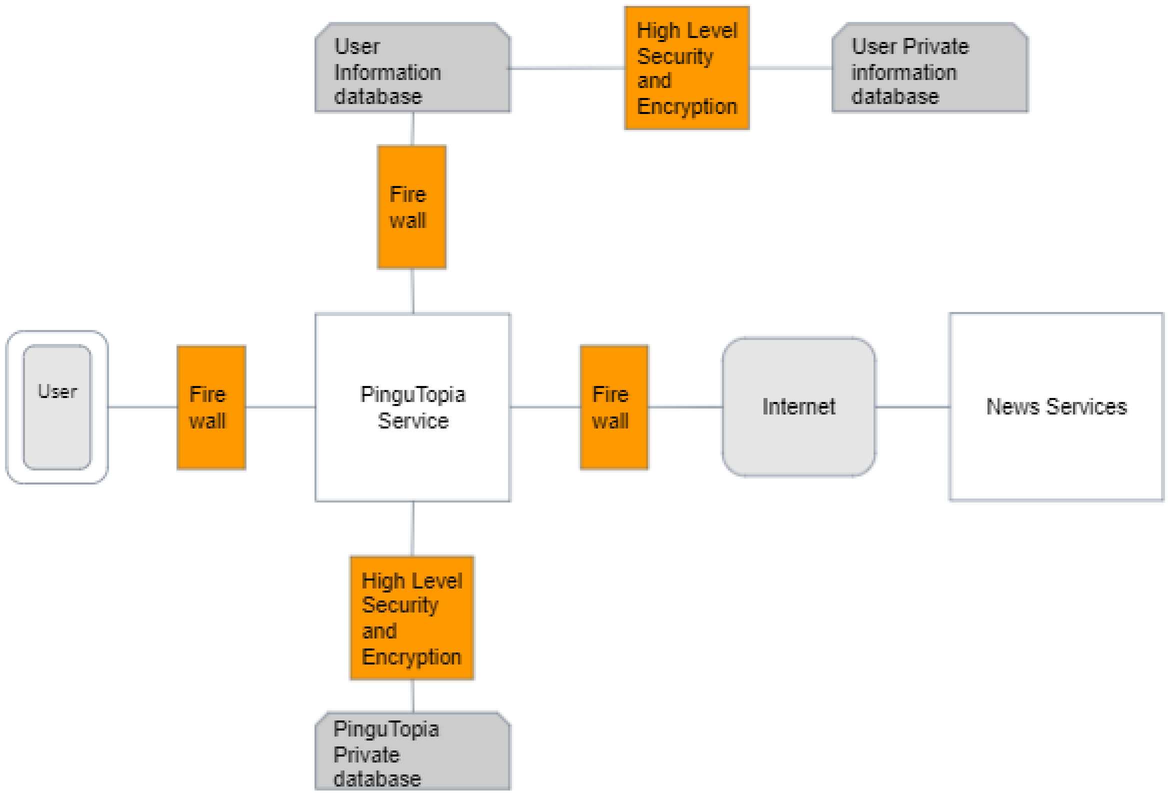
The variability can come from both diagrams. The news could be extracted with different methods that involve live weather then article filters wouldn't be needed. The way that the builder and modifier could be defined so that they are the same thing and the process could be done with different techniques such as the changes being made there and then the Empire can be the same source of storage of data. There could be a way the user interacts with impact so the user can prevent changes before they happen.

3.2.3 Architecture Background

The sequence diagram was chosen so I can display the processes involved in PinguTopia with clear and understandable structure. A sequence diagram is used to show how messages are sent between objects and the structure between those objects. This was proven as I did not need to explain too much about the elements because the diagram speaks thousands of words so it just needs a brief summary on how it all comes together. The reason for the two diagrams was because when I had it all as one diagram it felt a bit cluttered and became unclear when the arrows started crossing over each other. With the two diagrams it creates clear structures that are different but are clearly a part of the same system.

3.3 Physical View

This network diagram shows all the security that is required for the users to know for sure that their data is safe and secure while using this application.


User Information database
High Level Security and Encryption
User Private information database
Fire wall
User
Fire wall
PinguTopia Service
Fire wall
Internet
News Services
High Level Security and Encryption
PinguTopia Private database

3.3.1 Element Categories

- **Users** are going to access this application through their mobile device but will need to go through some firewall to ensure that they aren't using a hacked or edited version of the app that is not authorised by us.
- **PinguTopia Service** is where the server for the app is running. The service gathers data from users and news services when required to perform actions. Examples of this is when a user logs in to the app the service needs to check if the information matches what is in the database before granting access.
- **User Information database** is where the users app data is stored. This information will include their Empire, penguin count and details of them and other information that relate only to the application. There is also a firewall before the PinguTopia can access it. This is so the public can not access other users' data and/or edit the data.

- **User Private Information** is secure behind high level security and encryption because it is the user's Name, Email and Password. This kind of information is only used for a set few incidents such as users logging in to the application.
- **PinguTopia Private database** is also behind high level security and encryption as this is the information of features that are in the test phase or anything about the application that should not be available to the public at this time. Developer version of the application will also be in there.
- **News Service** is where the information used in the impact class is collected and PinguTopia Service will access this information through the internet. This information includes articles of relevance to Antarctica that can be used to impact the user's Empire.

3.3.2 Variability Guide

The main variants here is that you can arrange the security and a way that is more cost effective but due to the lack of knowledge I have in this area this was the best I could come up with. The data on the user information can be divided into Empire and Penguin as that when the PInguTopia service requires information on a user's penguins it does not need to access all the information. The type of security that is used could be changed but once again I do not have a large understanding in that field.

3.3.3 Architecture Background

The network diagram was used as this was the diagram that would provide information that was not already displayed in the previous views. Network diagrams let you see how each element shares information with another. With this you can see clear communication with the elements and create a vision on how the segments are bound to each other. This provides key information on how the software and the hardware worked together to make a smooth running application. With the lack of background knowledge on this subject I feel I have provided enough information that can be used to develop and enhance the project to be successful.

4. Conclusion

I feel that this architecture is a good start towards a successful product but just needs some improvement in the areas of weakness where I did not as much understand of, this being the network diagram. All the aspects and elements that have been enhanced have a wide variety of variability which will make development an easier experience because it gives the team options for any issues they run into while maintaining the core purpose of the product.

5. References

Smith, G. (2000). Goodbye North Pole? *Earth Island Journal, 15*(4), 24-24. Retrieved August 13, 2021, from http://www.jstor.org/stable/43880911

Anubha Sharma, Manoj Kumar, Sonali Agarwal, 2015. A Complete Survey on Software Architectural Styles and Patterns, Procedia Computer Science https://www.sciencedirect.com/science/article/pii/S187705091503183X

Wojcik, R., Bachmann, F., Bass, L., Clements, P., Merson, P., Nord, R., & Wood, B. (2006). AttributeDriven Design (ADD). https://resources.sei.cmu.edu/library/asset-view.cfm?assetid=8147

Philippe Kruchten, 1995. Architectural Blueprints—The “4+1” View Model of Software Architecture. https://www.fing.edu.uy/inco/cursos/ingsoft/iis/files/Pbk4p1.pdf

Kevin Curran, Ciaran George, 2012. The Future of Web and Mobile Game Development. https://www.researchgate.net/profile/Kevin-Curran/publication/273802365_The_Future_of_Web_and_Mobile_Game_Development/links/5717fbf208aed8a339e5b26b/The-Future-of-Web-and-Mobile-Game-Development.pdf