CHAPTER 2
THEORETICAL FOUNDATION

2.1 Theoretical Foundation

2.1.1 Social Media

According to the Oxford Dictionary [4], social media is defined as “websites and applications that enable users to create and share content or to participate in social networking.” A study conducted by Drahosova and Balco [5], with 275 EU respondents, found that social media websites are the fourth most-used websites when comparing with other websites such as search engines, video-sharing websites, and wiki-systems. These respondents believe that social media has many benefits such as the exchange of information and communication, the ability to share data, the ability to work from anywhere and still be able to communicate with co-workers, and many more. However, these respondents also believe that social media can also create problems such as internet addiction, users feeling a lack of security, and even information overload.

What used to be a way of simple and fun communication between friends, can now be used as tools for business. A study was conducted by Kumar and Pandey [6], with the purpose of the study to find out the importance of implementing social media as a business’ marketing tools. Their results show that companies who implement social media into their marketing strategy has a higher level of customer satisfaction. This is because it is simple for the company’s marketers to communicate with their target customers at any time. Social media has also created a two-way communication system instead of the traditional one-way system. This is because customers are now able to easily communicate their issues or problems to a company representative easily.
Implementing social media into a company’s marketing is also more cost-effective as social media advertisements are cheaper than television advertisements or billboards while increasing the company’s visibility to a larger number of customers.

These days social media has become an important part of our daily lives, this is especially true in Indonesia, with 198 million citizens using social media, making it the country with the third highest number of social media users.

**2.1.2 Online Video Platform**

According to Intricately [7], OVPs or Online Video Platforms, are “platforms for providing live and recorded content on a website.” Online Video Platforms allows users to upload content produced by the users themselves while also allowing other users to view or stream their content. OVPs are the ones that handle the hosting and managing tasks which means that users did not have to handle the programming or hardware for the video upload. One of the most famous OVP is YouTube.

According to Technopedia [8], YouTube is “a video sharing website that allows registered users to upload and share videos with anyone able to access the site.” Begun in 2004, as stated by Mohsin [9], YouTube has rapidly grown with 2.3 billion active users in 2021 watching billions of videos every day, 79% of internet users also state that they have a YouTube account. Google saw this website’s potential, and within a year after it first began, Google acquired it.
YouTube, as shown in Figure 2.1, started as a simple video sharing website but now it has been updated with many new features such as shorts. As shown in Figure 2.2, Shorts is a vertical video format that allows users to create 15-seconds or less videos, and live streaming. YouTube is now used for more than just watching content, it is used as a source of education. A study conducted by Aniroh, Ariyanto and Latifah [12], tried to find out how effective communication can be when a class is taught through YouTube live streaming. The result states that YouTube live streaming is effective communication wise, when used as a teaching tool, this is due to the practicality of the platform and enjoyable feeling for students and teachers, although the study also states that there are many drawbacks to implementing YouTube live streaming into a class such as noise or unstable internet, this was overcome by
implementing notes into the live stream chat box. Students participating in this study also felt that YTL should be implemented more, as students are able to study alone, at their own pace while also being able to repeat and watch the entire class at any time.

Another popular online video platform is Twitch, shown in Figure 2.3. According to Investopedia [13], Twitch is “a platform that allows gamers to be broadcasters, viewers, and participants in online gaming communities.”

As shown in Figure 2.4, and according to Garretson, Hamilton, and Kerne [14], a live video or “stream” on Twitch is a user or “streamer” producing live video and audio content with other users, known as “viewers” watching the stream able to communicate with the streamer through text chat boxes. Streams on Twitch are mostly streamers playing video games individually or with a group of friends. These streamers also implement a video of themselves while playing the game to make sure that the
stream feels more engaging and interactive as viewers are able to see how the player reacts and moves while playing a game. There are also other types of live streams on Twitch. Some streamers’ content are them communicating and interacting with their audience only, cooking with their audience, hosted game shows or other type of live content.

Although the general public thinks of Twitch as an entertainment source only, Twitch has proven that as a platform, it can also be used for other purposes. For example, a study conducted by Dombrowski, Faas, Miller and Young [15], observed creative streams on twitch to find out twitch’s effects on mentoring communities. These communities are usually made when streamers are able to help viewers learn and improve, which draws more viewers into the stream and a community is made. The study’s results show that mentoring is not done by the streamer alone, this is because viewers are also able to comment and educate about certain details that might educate not only the streamer, but other viewers as well. The results of the study also states that this community is not limited to Twitch as a platform, they also have chat servers that might discuss certain topics or details that could not be covered or explained during the stream. These results means that Twitch can also be used as an education tool.

2.1.3 Live Streaming

The Cambridge Dictionary [2], defines live streaming as “a broadcast of the video and sound of an event over the internet as it happens”. Live streaming allows users to watch, create and share videos in real time. According to Meisfjord [16], The first live stream ever broadcasted on the internet was a performance by Severe Tire Damage in 1993. This broadcast used Mbone which was a system designed to decrease
the bandwidth required to broadcast audio and video. Nowadays the general public can easily view live streams and even live stream themselves. This is because there are many live streaming platforms on the internet today, such as Twitch and YouTube.

This is a brief explanation of how live streaming works based on information from Cloud Flare [17]. The way live streaming works is separated into multiple steps. First, it starts with a video recording which is raw video data, this data is then compressed which means unimportant visual information is removed, and encoded, which means the raw video data is converted into a digital format that can be processed by many devices. Next, as the encoded video file stores a lot of information, the video file is separated into smaller chunks to send out the data faster.

This process is called segmentation. Next, the video is distributed through a CDN, as implementing a CDN means faster performance due to user requests being handled by a nearby server instead of the original server. This also reduces the original server’s workload, and as a CDN has many servers located across the globe, it also means the live stream is accessible anywhere. A CDN also saves each segment of a live stream temporarily so users can watch the live stream through CDN cache instead of the original server which makes the live stream nearer to real time. Lastly, the CDN send video data to users who watch the stream and each user’s device receive the video data, decodes the data, decompress the data and the video player within the device displays the video.
2.2 Theoretical Frameworks

2.2.1 Web App

According to Indeed [18], a web application is a program that is called or executed on a browser to solve a client’s specific problem. This program is split into two different sides, the client side and the server side. This client side is the program that is used by the user to run the web application, enter data and create requests while the server side is the application that stores and processes the data that is entered by the user on the client side and also handle all requests created by the user through the client side.

AltexSoft [19] states that web applications are typically divided into separate layers, which allows developers to work on each layer independently so other layers of the application will not be affected. There are typically three layers to a web application. The presentation layer, the business layer, and the persistence layer. However, applications can also have extra code not belonging to the layers, these are cross-cutting code and third-party integrations.

The presentation layer is the layer that allows the user to operate the web application. This is done through the user’s web browser and is the layer that contains user interface components, usually created using HTML, CSS and JS, that are used to interact with the web application. The business layer is the layer that accepts and handles the requests that are sent by users from their browsers. This layer is where the developers specify how the data and requests that the user asks for is sent to the back end of the web application.
Lastly, the persistence layer, is the layer that accepts all requests for data from the user and provides users access to the applications own persistent storage. This layer is very attached to the business layer as the logic for data requests are in the business layer and allows the persistent layer to know which database to interact with while also making a more optimized way of taking requested data. A typical web application’s data storage infrastructure will have a server, a database management system, software that allows communication with the database and a user interface to obtain and read requested data. Data can be stored on a data center or company owned hardware servers.

Cross-cutting code are modules that handle application concerns that are not handled within the 3 layers, these concerns could be security, communications, operation management, etc. These modules affect the entire web application but should not be put within the same layer.

Third-party integrations are code that is connected to the back end of the web application by small chunks of code called APIs. These APIs allow the web application to take data from other software and also increase the web application’s functionality. Some examples of third-party integrations are Payment gateways and social logins.
Figure 2.5 shows a simple explanation on how a web application works. First, a user makes a request to the web server through a browser by typing in the specified URL of the web application that the user wants to access. The browser sends this request to the Domain Name Center, a specific repository filled with domain names and their linked IP addresses, then the browser uses the HTTPS protocol to send a request to the IP address that is linked to that URL within the DNS.

Next, the web server who is using that IP address catches the request that is sent by the user's browser and this request is sent to the server's storage area to find the specific webpage and data that the user has requested for. The specified webpage and data are found by the web application’s Business Logic, this is because the Business Logic is the module of the application that handles how data within the server is
accessed. As the Business Logic handles the request, it notifies the web application to find the data that the user has requested.

Lastly, after the specified data and web page has been found, they are sent back to the user and they are able to see the web page that they requested for within their browsers [19].

There are many benefits of using a web app, a web application does not have to be installed onto a hard drive, so users do not have to worry about using up their storage space. It also requires less support and maintenance from the developers or companies who created them which leads to less development cost while also allowing slower electronic devices to run them which means more users have the ability to use the application. Web applications are also never out of date as all users access the same web application, so no single user is using an outdated version of the web app. Accessing the same application also means that there are no issues regarding compatibility between users as well. Web applications can be accessed by users anywhere on any device that has a web browser and an internet connection, with no regards for each user’s operating system, and lastly because no software is downloaded, web applications also reduce the piracy of software [18].

Some examples of web applications are Google Docs, a web application that allows multiple people to work on a document together in real-time. Facebook, a web application that allows you to communicate with anyone in the world anywhere. Dropbox, an online web application that stores data that users upload onto the cloud meaning that data could be accessed from anywhere and many more [18].
2.2.2 MVC

According to Tutorials Point [21], MVC or the Model-View-Controller (MVC) is “an architectural pattern that divides an application into three components: the model, the view, and the controller.” Figure 2.6 shows a simple diagram of how MVC theoretically functions. The Model contains all the logic for data that is interacted by the user. This can be data transferred between the View and Controller or any data that is interacted with by the business logic of the web application. The View is all the applications logic for the user interface. Lastly, the controller is the interface between the model and view, this means that the controller handles all the business logic and requests of the web application. Pitt [22], states that the main benefit of MVC is that it defines strict containers for all of an application’s code and features. This means if a developer makes changes to code within a single component, the other components will be safe from changes that might break the code.

2.2.3 Front-End Web Development

According to Lindley [23], Front-end web development is “the practice of producing HTML, CSS and JavaScript for a website or web application so that a user
can see and interact with them directly.” The main goal of developing a web application’s front end is to make sure that the relevant information the developer is trying to show to the user can be seen easily. This can be difficult as over time, users may access the web application through multiple, different devices that uses different screen resolutions and sizes which means that the developer must consider these changes when designing the front end of the web application. Developers need to guarantee that the web application appears correctly even when accessed through different devices, operating systems and browsers.

Front end web development is important because users are looking for applications that are easy to use yet have robust and detailed features. Results taken from a study done by Brown, Dudek, Fernandes and Lindgaard [24], shows that it takes 0.05 seconds for an average user to form an opinion of a B2B company by looking at their website. This means that a web application’s front end has the ability to increase a business’ chance of success. So, proper front end design, functionality, and experience is proven to be an important part of a web application.

Rishabh Software states [25], that the main points of proper front end design are Performance, Intent, Navigation and Retention. Performance means that applications that load faster will lead to an increase in user engagement and retention, the faster the loading time, the higher the increase. Intent means that the front end of the web application should be created in a way that aligns with the purposes of the business, this means that the design should not be confusing or misleading, instead it should be designed in a way that shows users relevant information regarding the company that answer questions that customers frequently ask. Good navigation means that users can easily find what they are looking for within the web application, this should be done
by implementing an organized and clean layout with proper graphics. Good navigation means that the users are able to use the application fast without any bloat throughout the process which could increase customer trust.

Retention means that the web application should have a user interface that is designed properly and can create trust within the user and allow the company to properly communicate their goal and message with the web application. This can help lead to an increase to user traffic and good retention will also highlight the web application’s main values to the user. When a web application has slow loading times, poor design and run-time errors, these points will reduce the customer’s interest to the web application and to the application itself, which means that the organization will have a bad reputation within the customer and less people will use the web application.

These main points can be implemented into the application by implementing proper branding, responsive design, and good interface design. Branding means that a web application with familiar content or design on the web application can help target the specific type of customer the organization is looking for; this can be done by using a consistent design language which can also promote similar types of products and services provided by the organization. Some characteristics that can create a consistent design language are the colors and fonts used within the web application, insight for the organization’s logo and lastly, a layout that is unique compared to other web applications.

Responsive design means that the web application is responsive to the user’s actions while also having a fast-loading speed. This means that the web application has smooth navigation through each module which can be used to see if an application will be successful or not. Some benefits of implementing responsive design to a web
application are first of all an improved experience for the users, increasing the user’s brand loyalty, creating a smooth and well-planned process for user purchases, and lastly increase an applications visibility while implementing lower development costs. Lastly, good interface design means that the web application has smooth communication for each activity the user can do. According to SWEOR [26], 94% of visitors form an opinion of the company’s credibility based on their visual appeal. This means that if a web application has a poor interface design, users will most likely not want to use it.

2.2.3.1 HTML

According to Tutorials Point [27], HTML or the Hyper Text Markup Language, is a formatting language that is utilized for defining the web page’s design and contents. This language allows developers to properly arrange elements such as text or pictures and videos that will be shown on the web page. In essence, HTML itself is not a proper “programming language”, it is instead a “markup language” that is used to communicate with the browser on how the web page should be displayed when a user opens it. This language is built on a number of elements that can enclose multiple sections of contents to display it in many different ways.

Mozilla [28] states that a typical HTML element will contain an opening tag, the content, and the closing tag. The opening tag will contain the element’s name which will have an opening and closing bracket, which will be used to specify an individual element. The content is the actual content that will be displayed within this element. And the closing tag is used to properly specify when an element ends. This tag will be near identical to the opening tag, except that the closing tag has a slash before the element name.
2.2.3.2 CSS

According to Tutorials Point [29], CSS or Cascading Style Sheets is used to help determine how a web browser will display the html elements while being a separate file. CSS can also be utilized to help change an html element’s specific formatting by making the changes in a single place and that these changes would be automatically be implemented for all instances of that element, even in different web pages.

Mozilla [30] states that CSS can be applied for either basic document text styling such as changing text colors or it can be applied to create layouts or animations. And according to, CSS is a “rule-based” language where the developer should define rules that detail style groups that should be applied to the HTML elements, singular or in groups.

2.2.3.3 JavaScript

According to Crockford [31], JavaScript is defined as the language of the web browser and is one of the most popular programming languages. JavaScript is lightweight and is used in web application development, where JavaScript allows client-side scripts to work with the user and create dynamic web pages. The most popular implementation of JavaScript is client side JavaScript, this version’s code is included or referenced in an html file so that the JavaScript code can be understood by the web browser.

Crockford [31] also states that JavaScript should be used because for one, as the internet is an important application platform and JavaScript is one of the only languages that is compatible with all browsers, web developers must have a strong knowledge of the language to be able to create great web applications. And the second
reason is that JavaScript is a great language that has many advantages, as previously mentioned JavaScript is lightweight, and according to Tutorials Point [32], implementing JavaScript means that there will be a reduction of server interactions, users will get feedback without reloading pages and also allows web developers to create impressive and functional interfaces.

2.2.4 Back-End Web Development

According to Hameed [33], Back-end web development is “the structure that permits the sharing of information across people and accounts.” Back-end developers are the developers whose mission is to make sure a web application works smoothly. These developers are the ones that focuses on code that handles the web application functionality and logic which is accessed or interacted with the user through the front-end. This means that whatever the back-end developer is working on, should never be seen by a user. Developing a back end is split between working on servers, applications and databases. Some of the tasks that a back-end developer would do are creating APIs that are used by the web application, creating and maintaining code that interacts with the database, making libraries of code, handling and improving the web application’s business processes and data architecture, and many more tasks. The key difference between front end and back end is that back-end developers work on how a web application functions, while a front-end developer work on how a web application appears to users. However, neither module works without one another, this is because the front end needs the back end to properly work, and the back end needs the front end for users to interact with the web application.

Codecademy states [34], that back-end web development is usually separated into three divisions, the server, the app and the database. In simple terms, the server is a
computer that is listening for requests from users. The application is the code that handles how a server should respond to the incoming requests and the database is the interface that is used to save data.

2.2.4.1 Python

According to Tutorials Point[35], Python is an interpreted, interactive, object-oriented, and high-level programming language. And according to Drake[36], Python is a simple yet robust programming language. It has data structures that are high in level while also being well ordered while being an easy implementation of object-oriented programming. Python is great for scripting or rapid application development as it has dynamic typing and a great syntax.

Navone [37] also states that Python is implemented in many different industries such as data science, machine learning, and web development. In data science, python can be used for data visualization such as plots or visual representations, in machine learning, Tensorflow is a free to use library that is implemented worldwide and is one of the most popular machine learning library. And for web development, python is implemented for the web application’s back-end functionality such as interacting with databases and making sure the application’s functionality run smoothly.

2.2.5 Django

According to Mozilla [38], Django is “a high-level Python web framework that enables rapid development of secure and maintainable websites.” Django was originally created between 2003 and 2005 by a team of newspaper website developers. This team factored out and reused many of the design patterns and common code that
they implemented into the newspaper websites, and it was evolved into the first version of Django, the web development framework.

Some benefits of Django are that it is Complete, Versatile, Secure, Scalable, Maintainable and Portable. Complete means that Django has everything that a developer might need built-in and this means this bundled functionality work effortlessly together, follow the same design principle, while also having detailed and updated documentation. Versatile means that Django can be used to create many different types of web sites such as wikis, social networks, and many more. Django is also compatible with any client-side framework while also having the ability to deliver content in multiple types of formats. Secure means that Django is built to make sure developers steer clear of security mistakes by making Django works by using good security practices while also protecting the application from external threats such as SQL injections, cross-site scripting, and many more. Scalable means that Django is implementing a component-based architecture which means that each module of the application can be upgraded or replaced independently. Maintainable means that Django is written in a way that motivate coders to create maintainable and reusable code. This is done by implementing DRY which reduces the amount of unnecessary code. Django is also built in a way where related functionality is grouped together.
Portable means Django is not limited to a single type of server platform or web hosting provider as it is written in Python.

Django is split into different types of files, Views, Models, URLs, and Templates as shown in a simple diagram in Figure 2.7. URLs are Python files that are used to redirect a HTTP request to a specified Django View file. A View file is a Python file that contains Django’s request handler function, so it receives the HTTP requests and sends out HTTP responses. Models are Python files containing Python objects that describe the application’s data structure and provides database functionality. Lastly, Template files are text files that define a file’s structure by using placeholders. A View file can create an HTML using a HTML template file.

2.2.6 Django Channels

According to Django [39], Django Channels is a project that aims to allow Django to use WebSockets, chat or IOT Protocols and more. Django Channels is based on the idea “turtles all the way down” which states that if a developer is able to use the standard Django techniques while also having access to an asynchronous interface for
According to Benfield, Mitchell, Samuel, Tardy, Warner, Williams and Zadka [41], the main point of Django Channels is to direct Django towards events rather than requests and responses. This means that applications that implement Django Channels are able to utilize WebSockets and other protocols rather than being limited to HTTP requests and responses. Django Channels work by separating the web stack into three components, An ASGI server, A “channel-backend”, and a worker. The ASGI server
handles all incoming connections while also translating write protocols to ASGI messages, which is then placed into queues, and also handles response from channels which is then translated back into wire protocol data. The channel back end is a data store that can also be utilized as a message broker. Lastly, workers are used to listen on some or all of the channels and if there are messages to process, the workers are responsible for running the code needed to process those messages. Figure 2.8 describes the Django Channels architecture.

2.2.7 Flask

According to Bekabil [42], Flask is a micro web framework that allows developers to create web applications with little to no difficulty. Flask can be used for any type of project with any type of scale. Flask has three main advantages, simplicity, openness, and well documented. Simplicity means that Flask is simple enough for first time developers to learn about the framework while using it. Openness means that Flask is distributed under a permissive open-source license, which means that it is more approachable and up to date. Lastly, Flask is regarded as well documented because it has a useful start guide, and revised documentation for each version of Flask.

2.2.8 OAuth

According to Fruhlinger and Grimes [43], OAuth is an open-standard authorization protocol or framework that describes how unrelated servers and services can safely allow authenticated access to their assets without actually sharing the initial, related, single logon credential. In 2010, OAuth was released to the general public as an open standard called RFC 5849 and it was heavily revised for the next two years, with OAuth 2.0 released in 2012, this time called RFC 6749.
Anicas [44], states that the way OAuth works is by delegating user authentication to the service that hosts the user account, and authorizing third-party applications to access the user account. For example, if a user is logged in into a website, and that website has a feature that interacts with a different web application or service, the website will, through OAuth, connect to the requested web application or service while also showing the details of the logged in user, next the requested web application or service generates a token and secret that is unique to this specific request and can only be used once. After this the website takes the token and secret and send it to the user’s browser. The browser will then pass on the token and secret to the browser’s authorization provider.

After the authorization process, the browser will ask for the user’s approval to interact with the requested web application or service. If the user approves, a new token will be generated and this token will be passed on to the website. And the website will pass on this token to the requested web application or service and the users requested process will start. OAuth is special because it can work across the web and it is implemented by many websites and services such as Amazon, Facebook, Instagram to name a few.

2.2.9 SDLC
According to Sharma [46], SDLC or Software Development Life Cycle, is “a very important software engineering concept that is utilized by developers to create a clear path for the thesis’ different phases to ensure the final product will be quality software.” As stated by Tutorials Point [45] in Figure 2.9, these phases include the planning phase, defining phase, designing phase, building phase, testing phase and the deployment phase.

The planning phase is where the SDLC process starts. First, senior members of the development team will communicate with the customer, the sales department, and experts relating to the customer’s field to learn about the product’s requirements and to also use information provided by these individuals to create a basic plan for the thesis, while also conducting a study on the software’s feasibility regarding economical, operational, and technical issues. This stage will also include steps such as preparing the requirements for quality assurance while also identifying the risks related to the software. The results of the conducted study will describe the methods that the team can take to create the product with the least number of issues.

The defining phase is when the development team create and describe the thesis’ requirements and get these requirements approved by the customers or other parties involved. This is done by taking the results of the planning phase, specifically the results of the requirement analysis, and write down the thesis’ needs in an SRS or software requirement specification document.

The next phase will be designing. This is done by product architect members of the
development team. They will take the SRS document and try to create and describe the architectures that are most suited for the thesis, these architectures are then described and written down in a DDS or design document specification document. This document will then be sent to all major parties involved which will, after considering each architecture’s risks, design modularity, budget and time restrictions, choose the best architecture for the product.

Next will be the building phase, this is where actual programming begins and the product is finally built. Code for the product will be based on the DDS specifications and the development team must stick to the company’s specified coding guidelines, and the programming language used for the product will be chosen with the type of application being made in mind.

This next phase will be the testing phase. Although there are many testing activities in all the phases, this phase is specifically used to ensure that the final product of this thesis will follow the SRS’ standard for quality and make sure there are no issues on the final application.

The last phase will be the deployment phase, this is the phase where the product will be released after it has passed through the testing phase. The release of the product itself will depend on the organization and after time and feedback, the organization might add new features into the product or create maintenance for issues that were not found on the testing phase.

According to Swersky [47], the benefits of implementing SDLC into a thesis are clear communication between the development team and parties involved, all members of
the development team have clearly defined roles, all inputs and outputs of each phase are documented, and each phase’s progress is clear therefore it will be easy to see if a step is fully finished.

### 2.2.10 Prototyping SDLC

![Diagram explaining the Software Prototyping SDLC](image)

Figure 2.10 Simple diagram explaining the Software Prototyping SDLC [48]

For this thesis, the author has implemented the Software Prototyping SDLC for the web application’s development. According to Galin [49], the Software Prototyping or Prototyping SDLC is an iterative methodology where a prototype of the final product is created at each iteration. The first iteration of the SDLC will start with a list of initial customer demands, and a developer will create a prototype that fulfills some of the demands which will then be shown to the customer for evaluation. The customer will then give comments regarding mistakes, revisions and things that the customer wants the developer to add to the next iteration. The developer then starts a new iteration, creating a new prototype, adding another one of the initial demands while
also including fixes and requests made by the customer during the previous iteration’s evaluation.

As shown in Figure 2.10, The main steps in the prototyping SDLC are divided into Communication, Quick Plan, Modeling and Quick Design, Construction, Deployment Delivery and Feedback. Communication is the first step in this SDLC and is the step where the customers give a list of demands for the developer to implement into the application. This is the start of the project’s initial iteration. The next step, Quick Plan, is the step where the development team takes the list of customer demands, and determines which demands are the most important. Next, the development team creates a plan for the initial prototype’s development, which will fulfill the demands that were chosen in Quick Plan. This step is called the modelling and quick design step. Next, the initial prototype is built by the development team in the Construction step. Lastly, the development team presents the initial prototype to the customer in the Deployment, Delivery and Feedback step. This step will determine additional fixes and demands that the user has for the final version of the application and the development team ends the initial iteration, and starts the next iteration with the comments from the user in the DDF step, while creating a new prototype that will be based on revising the original prototype. This process will be done again and again until the customer is satisfied with the prototype of the application. If the customer is satisfied with the prototype, the development team will then create the final version of the application based on the approved prototype.

According to Susanto and Meiryani [50], The main benefit of implementing Software Prototyping as a SDLC is that there are clear lines of communication during the
development process of the application between the customer and the developer, customer needs are also clearer as the customer themselves have an active role in the development process, and final version of the application is ensured to fulfill the list of customer requests as well as new requests as customer requests are repeatedly discussed in each iteration.

Also, Tutorials Point [51] states that there are many types of Prototyping, including Rapid Throwaway, Evolutionary, Incremental, and Extreme. Rapid Throwaway Prototyping is a version of Prototyping where the prototype is based on the initial demands, and shows how the application will look, the customers will then evaluate this prototype, make changes to the initial demands, and a new prototype will be created by the development team based on the revised prototypes. In Rapid Throwaway, not all prototypes will be used in the final design, as some prototypes may have mistakes or faults that the customer does not want in the final application. Evolutionary is a version of Prototyping that creates an initial prototype based on the initial demands, and this prototype will be more refined during each iteration of development because the demands and requests of the customer during evaluation will be done by developers on this prototype instead of creating a brand-new prototype for each iteration.

Incremental Prototyping is when the development team takes the list of initial customer demands, and each demand is worked on and prototyped separately, which will then be merged together into the final application. This reduces the development’s complexity as each demand will be divided and developed in a more efficient way. Extreme Prototyping is a web development only version of Prototyping. There are
three phases in Extreme Prototyping, the first step is creating a basic prototype with static html pages. The next step, a functional prototype using artificial processes and lastly creating a final prototype with real functionality. This method ensures that the development team are focusing on the specified demands instead of focusing on unnecessary things.

2.2.11 User Interface

According to Galitz [52], User Interface is defined as the segment of the computer software that users can understand or direct by using the 5 senses. There are two components in a user interface, the input and the output. Input is defined as how the users are able to tell the software what the user wants it to do. An example of a user interface input is a keyboard or a mouse. The next component is output which is defined as how the software is able to show the results of what the user requested. An example of a user interface output is a screen or speaker.

A well-designed user interface will use inputs and outputs that will solve the user’s requests effectively. A good user interface is important as it is how the user is able to view and access the software, while also being the segment that users see when a software completes their request. If the user interface is improperly designed, it will be difficult for users to complete their tasks properly and efficiently. Badly designed user interfaces can even reduce the amount of people using the software.

2.2.12 User Experience

According to Benyon [53], UX or User Experience, is defined as the experience that is felt by users when using the software. UX is different from UI as UI is usually scored or tested by using usability testing or quantitative measurements. UX however
is tested by viewing how users feel and think while using the software. The main goal of proper UX design is to make software enjoyable and favorable for users minimizing bleaker or pessimistic emotions while using the software. This is important as according to, Proper UX design can create user loyalty to the software.

![Diagram explaining 4 steps of UX Design](image)

Figure 2.11 Diagram explaining 4 steps of UX Design[45]

There are four steps when designing a proper UX design. According to and as shown in Figure 2.11, these steps are Understanding, Design, Envisionment, and Evaluation. Understanding is the step where a designer learns what the software has to do, and what the process has to feel like for the users, this is usually done by researching a lot of potential users and targeting the needs learnt from the research and try to create that experience with available technologies.
Next, Design is split into two steps, conceptual and physical. Conceptual focuses on discovering what information and services are needed to achieve the user’s goals. It focuses on clarifying the design concept of the software that will be displayed to users. Physical design focuses on taking the points learned from conceptual design and creating actual physical design based on these points. It focuses on what technologies the software will require and also what the users will need to know to use this software.

Next, Envisionment is the step where the designers focus on finding the proper media to render or display ideas that were made in the design step. The designer must consider the users, the available technologies how far along the development of the software is while trying to follow the designs created in the previous step and making sure that issues are shown clearly.

The next step, Evaluation is near to Envisionment as the media used in Envisionment will affect how it is evaluated. Evaluation criteria will also depend on who is able to use the representation. Any of the other design activities will be followed by an evaluation. Sometimes this is simply the designer checking through to make sure something is complete and correct. It could be a list of requirements or a high-level design brief that is sent to a client, an abstract conceptual model that is discussed with a colleague, or a formal evaluation of a functional prototype by the future system users. Techniques for evaluation are many and various, depending once again on the circumstances. Expressing the design ideas in terms of a concrete scenario that people have to work their way through can be very effective. The important thing to keep in mind is that the technique used must be appropriate for the nature of the representation, the questions being asked and the people involved in the evaluation. Later in the design
process, designers can evaluate alternative designs using A/B testing and find out what different users are doing using web and app analytics. According to Siroker and Koomen [54], A/B testing is a form of testing where developers show test users different versions of their program and determine which version is best at attracting customers.

2.2.13 Figma

According to Bracey [56], Figma is an online software that is used to develop user interfaces and prototyping, while also allowing users to organize and collaborate. Figure shows an example of Figma being used to design a user interface. Kopf [57] states that the main benefit of Figma is that it allows multiple designers to collaborate remotely, while also allowing users to collaborate through any device that has an internet connection and a web browser on its operating system. Figure 2.12 shows an example of using Figma to design user interfaces.

2.2.14 Shneiderman’s 8 Golden Rules

According to Cohen, Diakopoulos, Elmqvist, Jacobs, Plaisant and
Shneiderman [58], Shneiderman’s 8 Golden Rules, or the 8 Golden Rules of Interface Design, are a set of rules created by Ben Shneiderman that is defined as a set of rules that are created from experience and is able to be implemented into interactive systems after it has been filtered, expanded and interpreted. The 8 golden rules are important to consider as they allow developers and designers to create high quality useful and satisfactory user interfaces.

The first rule is always strive for consistency, this rule states that elements such as language or commands should be kept consistent. The second rule is enable frequent users to use shortcuts. This rule states that for users who have used the application multiple times, the application should have features that speed up user actions. This is usually implemented by using hidden commands or function keys.

The third rule is offer informative feedback. This rule states that the application should inform user about what they have done for all actions the user did. The fourth step is Design dialog to yield closure. This rule states that the application’s functionality’s flow should be separated into beginning middle and end sections while also informing user if the flow is completed.

The fifth rule is offer simple error handling, this rule states that developers must make it so that no major errors can occur from user interaction, and to display solutions for any user-based error. The sixth rule is permit easy reversal of actions. This rule states that the software must inform users that errors can be undone.

The seventh rule is support internal locus of control, this rule states that the application
should be designed in a way where users feel that their decisions are what is responsible for their actions instead of responding to the application which leads to a feeling of being in charge. The eight and last step is reduce short-term memory load. This rule states that the developer should minimize information shown to users which can be done by implementing simple designs, reduce the amount of movement within the application.

2.2.15 Firebase Cloud Messaging

According to Moroney [59], FCM or Firebase Cloud Messaging is a messaging solution that lets you send messages to multiple platforms that is also designed in a way that is reliable and scalable, as Firebase Cloud Messaging having a 98% successful message delivery rate in under 500ms while also being able to deliver trillions of messages each week.

![Firebase Cloud Messaging Diagram](image)

Figure 2.13 Diagram explaining Firebase Cloud Messaging steps [60]

Figure 2.13 explains the Firebase Cloud Messaging architecture. The first component is the tool used to create message requests. This can be the notifications composer if
only notification messages are needed or a trusted server environment that can utilize Firebase Cloud Messaging protocols and the Firebase Admin SDK. Some examples of a trusted environment server are Cloud Functions for Firebase or App Engine. The next component is the FCM back-end, which is able to accept message requests, and create metadata for each message.

The third component needed is a platform-level transport layer, this is used to direct each message to the specific device they want to reach, control how each message is sent, and implement necessary configurations. Some examples of are the Android Transport Layer, the Apple Push Notification service and the web push protocol. The last component is the FCM SDK that is on the targeted user’s device, where the message is displayed and handled depending on the configuration of the message.

A study conducted by Albertengo, Fikru, Hassan and Stramandino [61] compared FCM to 3 other message sending platforms in terms of delay, data efficiency, and power consumption. And the results from their study found that Firebase Cloud Messaging is viable if the data transmitted is small or when mobile applications require FCM features.