Project Plan

for

Streamify

Prepared by: Avinash Ratnam (PES1201800883),

Harichandana Magapu(PES1201801041),

Utsav Jolapara(PES1201801086)

PES University

20th February 2021

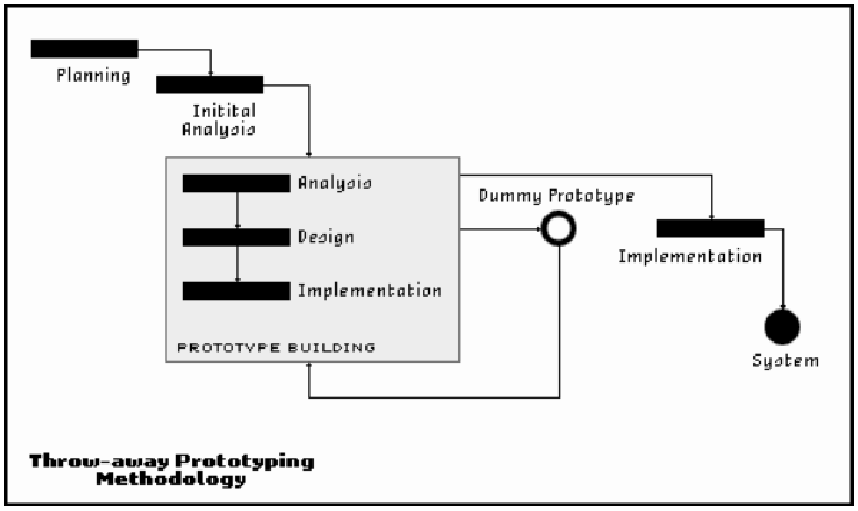
I. Software Development Life Cycle Approach:

Based on our understanding and our product, we deduced that the **‘Prototyping Model’** would be the best fit. This is due to several reasons as given:

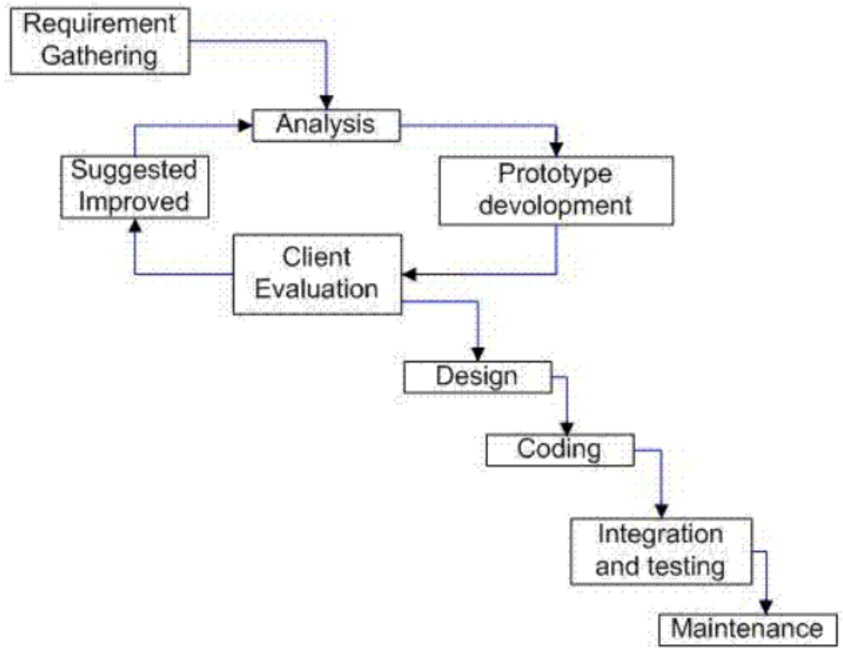
1. The Prototyping Model allows for visualising the product, reducing the misunderstanding between the developers and the customer.
2. It reduces the number of iterations made to the product, making it much more flexible than traditional linear development approaches such as the Waterfall method.
3. Since our product would be incomplete due to updates to the app (ideally), the Prototyping method allows for changes to the product and merging it with another prototype.

Different types of Prototypes can be created:

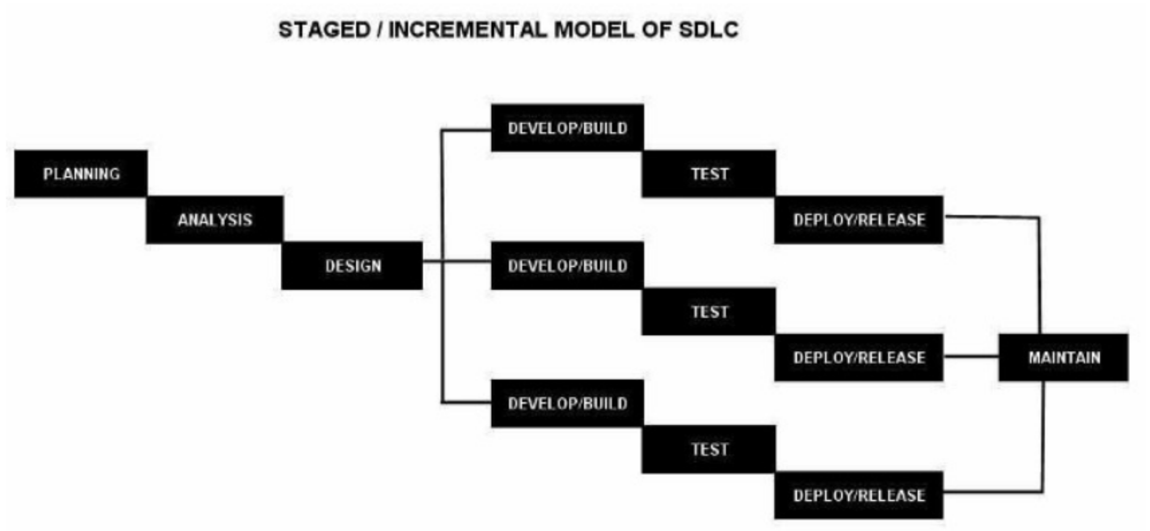
1. **Throwaway prototyping**: Prototypes that are eventually discarded rather than becoming a part of the final delivered software.
   Egs:



2. **Evolutionary prototyping**: Prototypes that evolve into the final system through iterative incorporation of user feedback.
   Egs:



3. **Incremental prototyping:** The final product is built as separate prototypes. In the end, the different prototypes are merged in an overall design.



4. **Extreme prototyping:** This is used in web applications mainly. It breaks down web development into three phases, each one based on the preceding one. The first phase is a static prototype that consists primarily of HTML pages. In the second phase, the screens are programmed and fully functional using a simulated services layer. In the third phase, the services are implemented.

Advantages

1. Reduced time and costs, but this can be a disadvantage if the developer loses time in developing the prototypes.
2. Improved and increased user involvement.

Disadvantages

1. Insufficient analysis. User confusion of prototype and finished system.
2. Developer misunderstanding of user objectives.
3. Excessive development time of the prototype.
4. It is costly to implement prototypes.

II. SDLC Tools:

1. **Planning Tool:**
   For collaborating and planning, we plan to use Discord. When working with Scrum teams, one challenge we see is that they are afraid of estimating velocity and all the other metrics. Hence, being in an environment they trust makes it easier to improve as a team.

2. **Design Tool:**
   For prototyping and wireframing the UI and the UX, we will be using Adobe XD. Adobe XD provides industry-grade features and design elements to help us create professional interfaces. XD also has a valuable part in simulating the interface in real-time to understand how all the components interact.

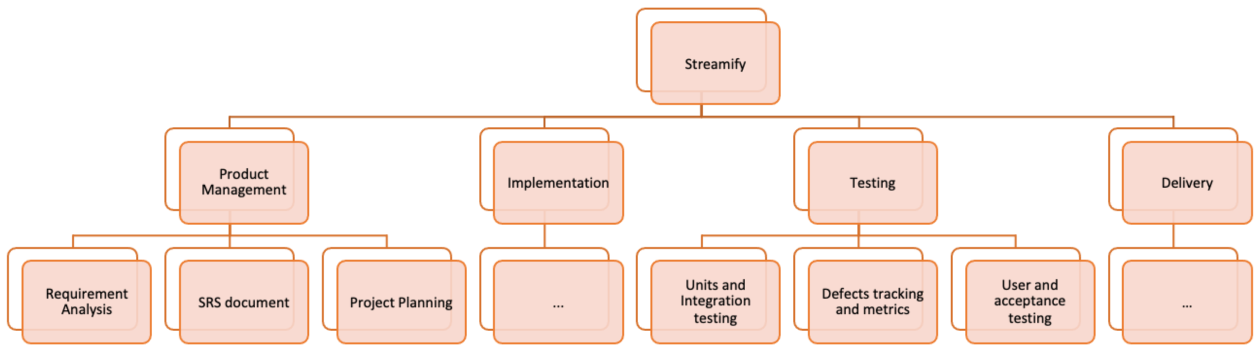
3. **Version Control:**
   Github Projects will be used to organise the workflow between developers and to track feature implementations. GitHub also helps in maintaining streamline development and ease of code review.

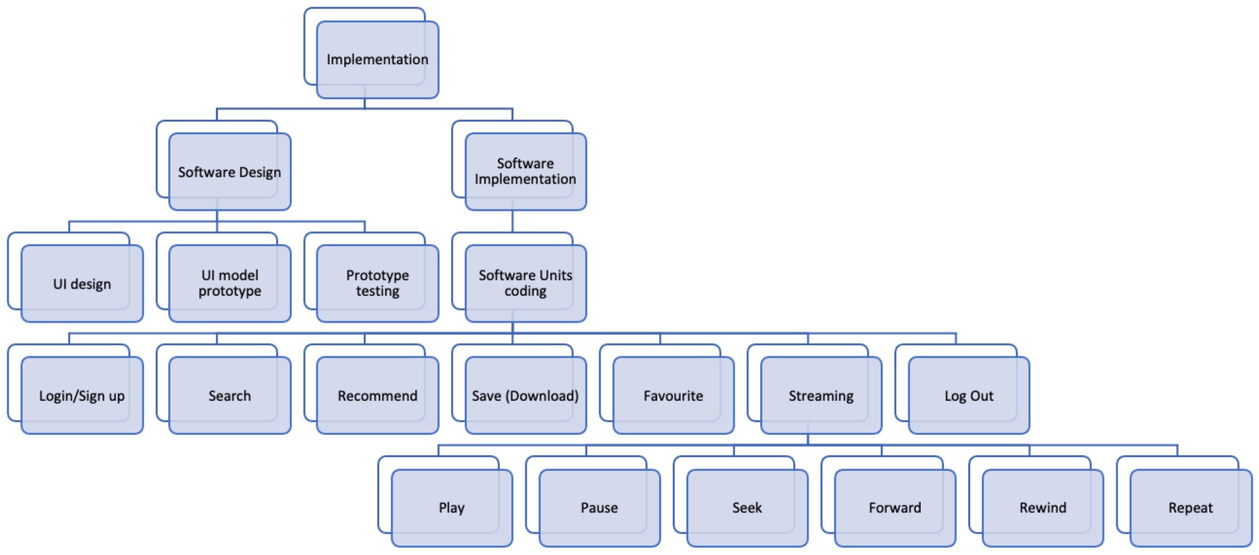
4. **Development Tool:**
   Visual Studio Code is chosen over other IDEs. It has inbuilt support for writing Node JS code and is easy to deploy the code onto git version control from visual studio code directly.
   Apart from VS Code, other IDEs such as Sublime Text will be used while coding as well due to its much cleaner UI and easy interface.

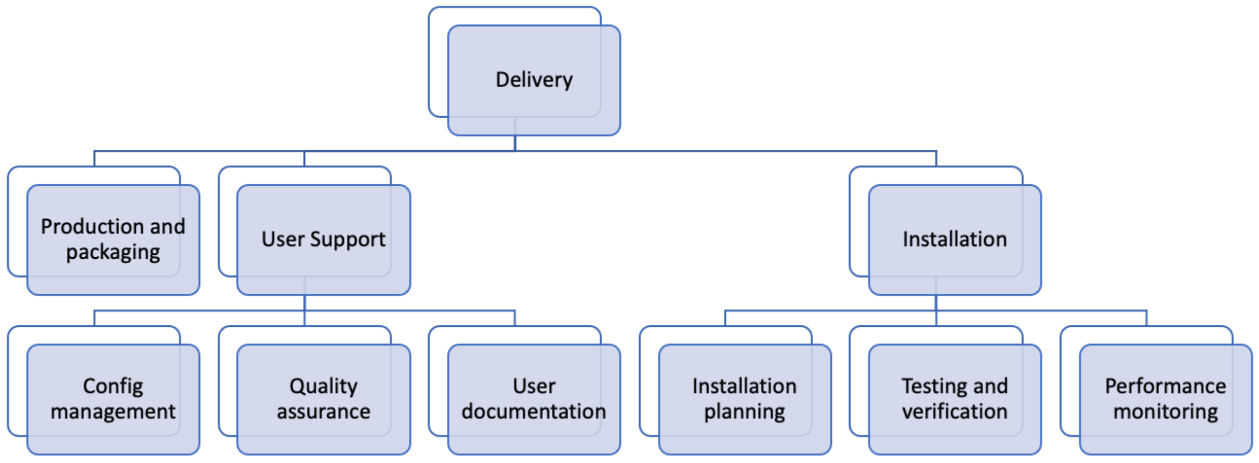
5. **Use-Case Diagram:**
   We will be using StarUML for the creation of all use-case diagrams.

6. **Gantt Chart:**
   We will be using LucidChart to create our Gantt Chart.

III. Work Breakdown Structure:







IV. Deliverables:

1. Software Design:

Component	Build/ Reuse	Description
High-Level Design	Build	Documentation of features provided by the application.
Detailed Design	Build	Documentation of the application features along with Action Diagrams and Use Case Diagrams.

2. Development:

Component	Build/ Reuse	Description
Front End Development	Reuse	Front end components, the UI and UX will be based off and inspired by existing services such as Spotify and Apple Music.
Back End Development	Build	The database to store the relevant information will be built from the ground up.
Integration Plan	Build	Combined using frameworks

V. An estimate of effort in terms of Person Months:

Constructive Cost Model (COCOM): COCOM is a regression model based on LOC, i.e. the number of Lines of Code. It is a procedural cost estimate model for software projects and often used to predict the various parameters associated with making a project, such as size, effort, cost, time and quality.

There are several types of COCOM's, but on further analysis, we concluded that our project was best apt to the **'Organic Model'**.
A software project is an organic type if the team size required is adequately small; the problem is well understood. It has been solved in the past, and also, the team members have little experience regarding the situation.

Some important formulae that are used to determine work and time are as follows.
This is irrespective of the model we chose, i.e. Organic Model.

Effort = a * (KLOC) ^ b, here KLOC is Kilo Lines of Code.

Time = c * (Effort) ^ d

PersonRequired = Effort/ Time

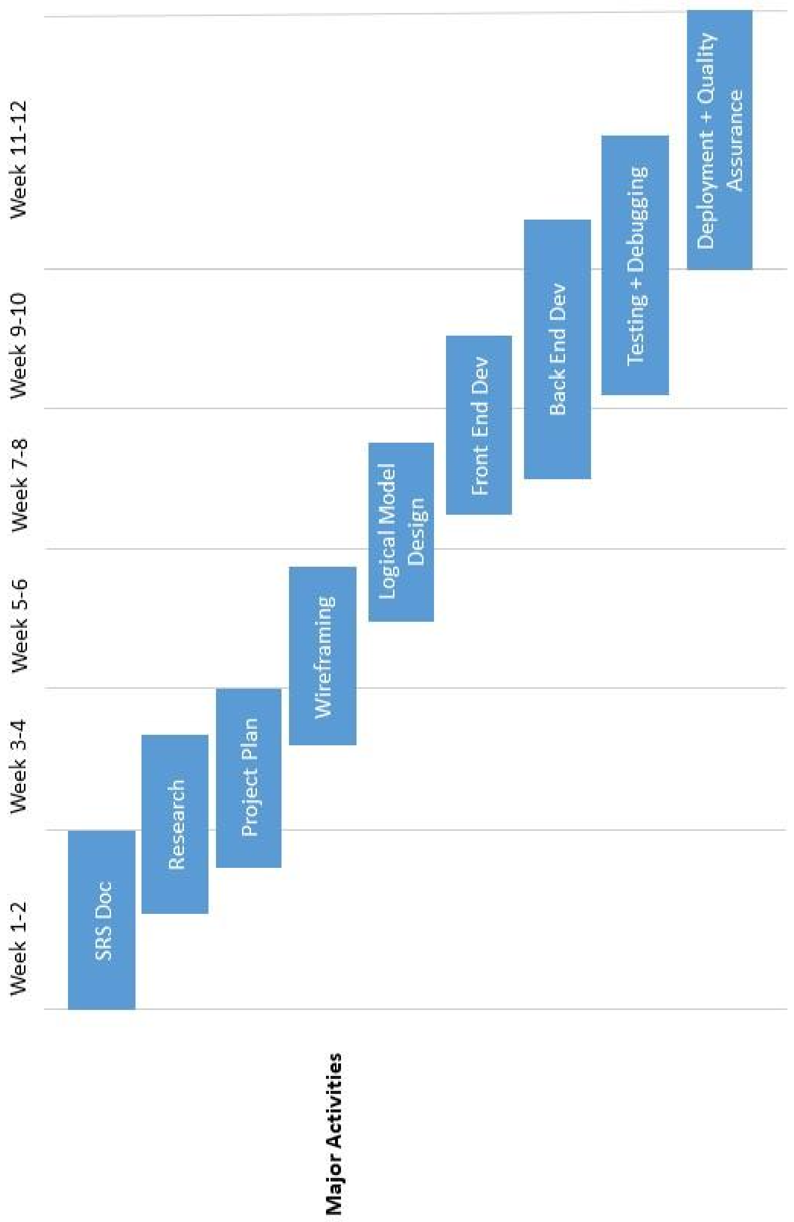
The constants a,b,c and d have the following values for an Organic Model;

a	2.4
b	1.05
c	2.5
d	0.38

Hence on substituting in our formulae, we have:

Effort = 4.96 PersonMonths
Time = 4.59 Months
Hence, **PersonRequired** = 1 (approx)

VI. Gantt Chart


Week 1-2
Week 3-4
Week 5-6
Week 7-8
Week 9-10
Week 11-12
SRS Doc
Research
Project Plan
Wireframing
Logical Model Design
Front End Dev
Back End Dev
Testing + Debugging
Deployment + Quality Assurance
Major Activities