Live Recorded Video:
https://youtu.be/mhuxYT8mv8w

Friendstagram

Make your university experience more fruitful

– *Team Inspiration*

Our Team

Manav Arora
U1822077D
Project Manager

Jovan Huang
U1921768B
QA Manager

Tan Hui Zhan
U1922013D
QA Engineer/
UI+UX

Royce Ang
U1840416D
Lead Developer

Clarence Hong
U1922950G
Release
Engineer/Manager

Zhu Weiji
U1922876G
Backend
Developer

Outline of Presentation

01

Product Introduction

Key features of the product

02

Design for Maintainability

Measures taken to ensure product is maintainable

03

Software Quality Assurance

Processes to ensure quality of product

04

Project Management

Breakdown of Project

05

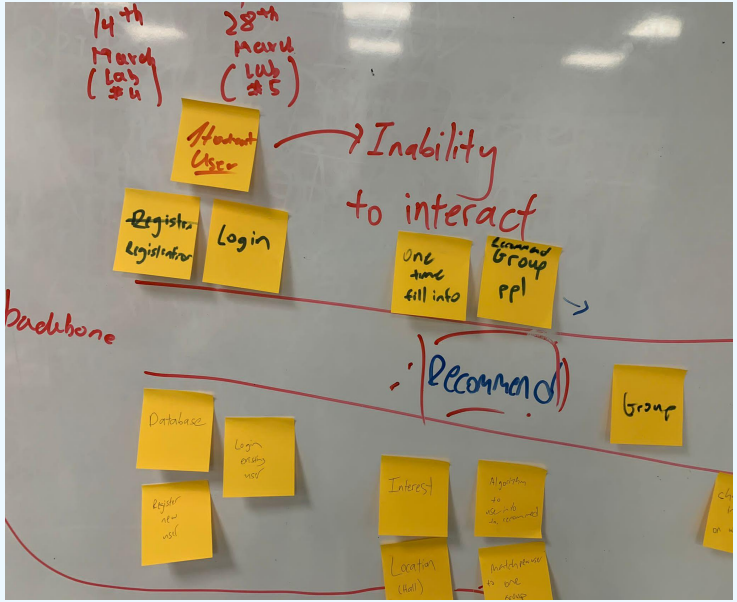
Risk Management

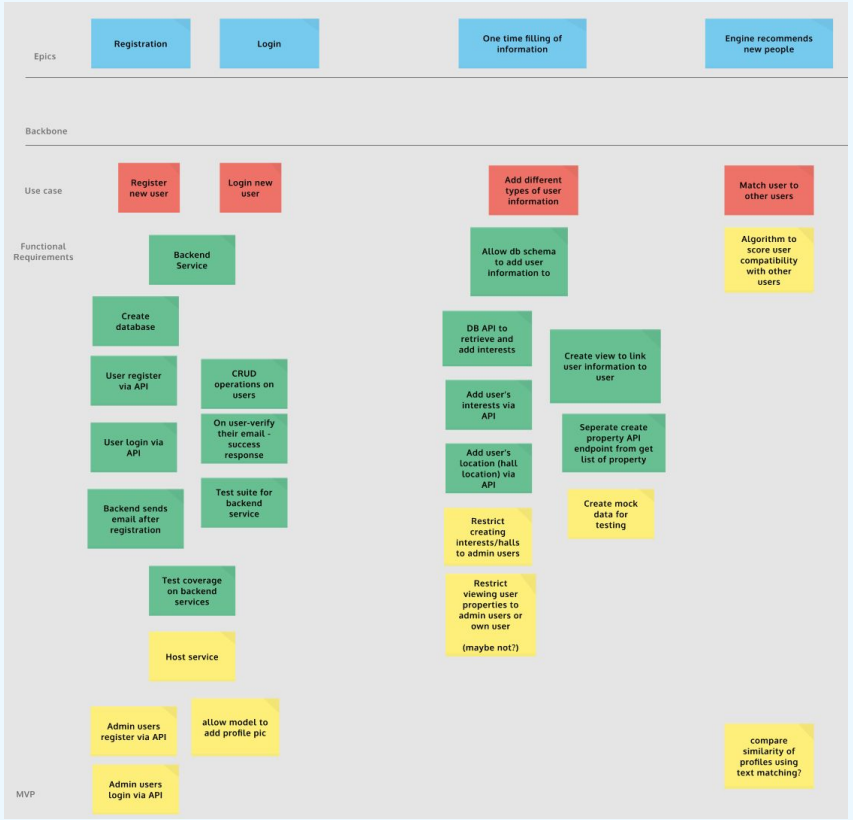
Reasoning, Approach and Risk Processes & Plans

01 Product Introduction

What are the current issues brought about by Covid-19 that you would like to solve as an NTU student today?

User Story Mapping - Jeff Patton





Pain Points To Resolve

Lack of interaction

Lessons become online & CCAs are cancelled

Fear of Socialising

Students aren't sure how to start rebuilding their social lives.

No sense of belonging

Students find it challenging to feel connected to a community

Unhappiness

Significant drop in satisfaction with courses in university

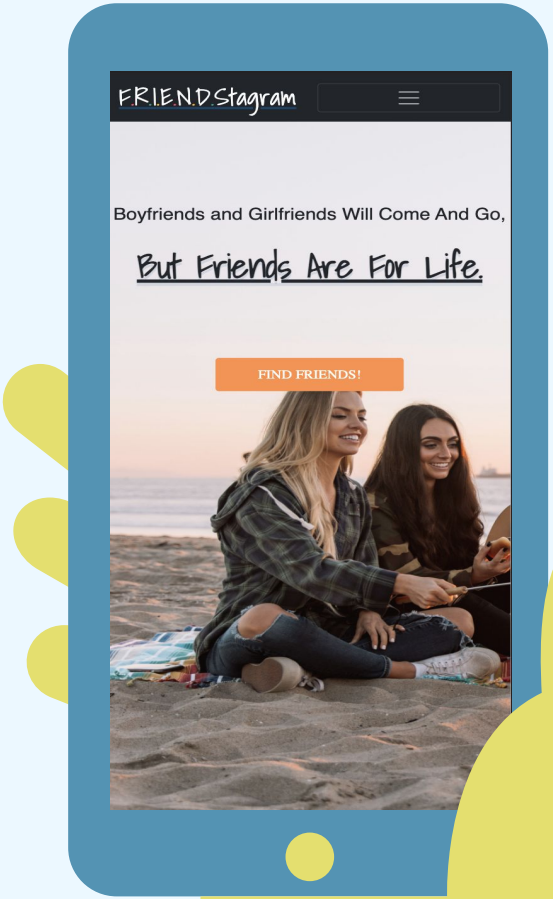
Aim

A personalized web-app that helps
Like-minded Students
Connect,
Find New Friends,
Rediscover Sense of Belonging

FIND FRIENDS

Feature #01

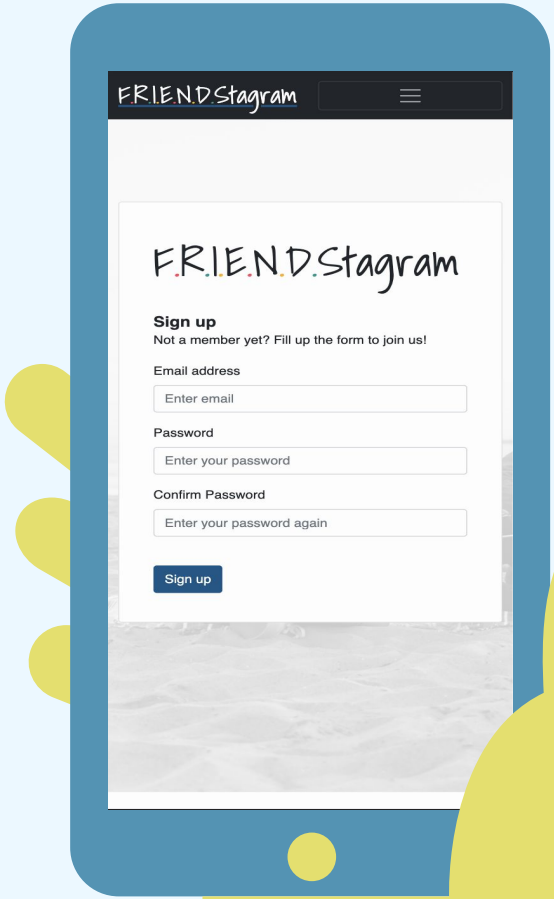
Friends recommended to the user through a carefully curated and highly personalised state-of-the-art matching algorithm



ACCOUNT CREATION

Feature #02

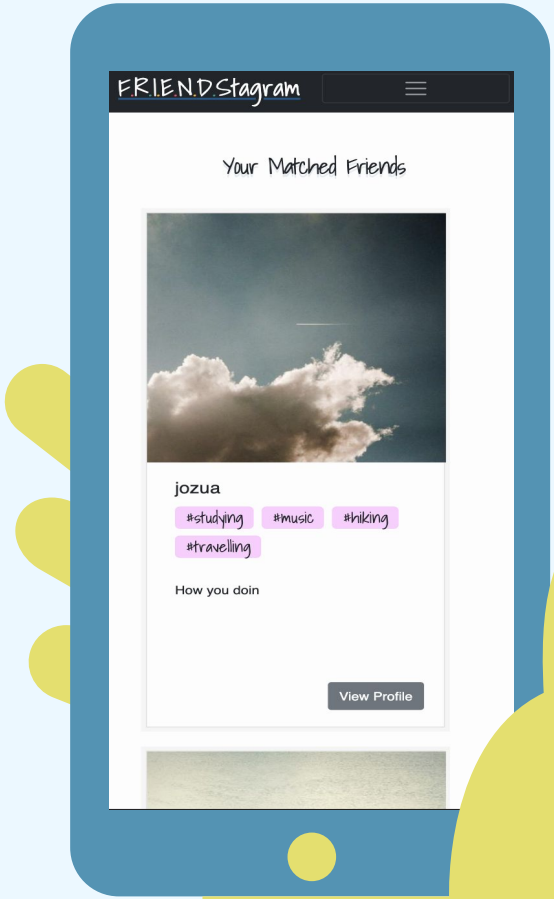
Allowing users to create their own accounts with interests and halls for a more personalised experience.



BROWSE MATCHES

Feature #03

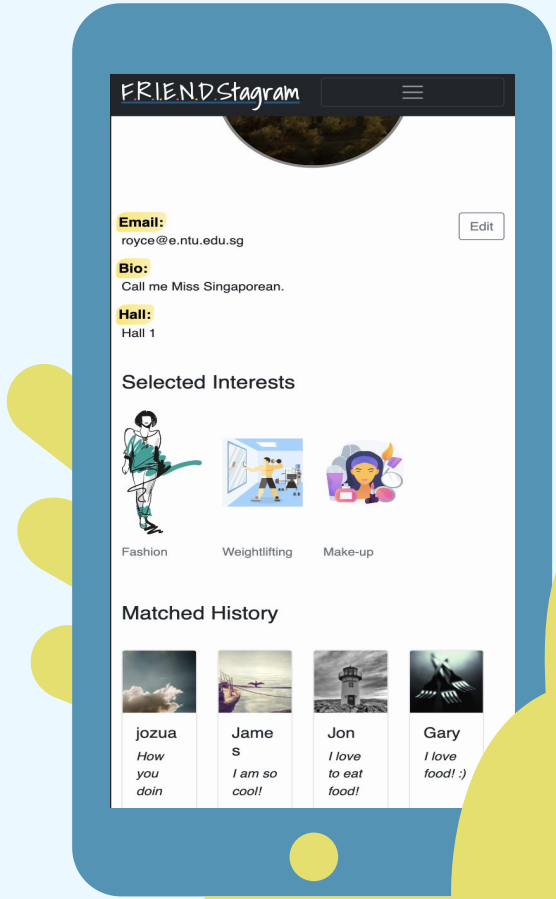
A catalogue of recommended friends that users can browse through.



MATCHED HISTORY

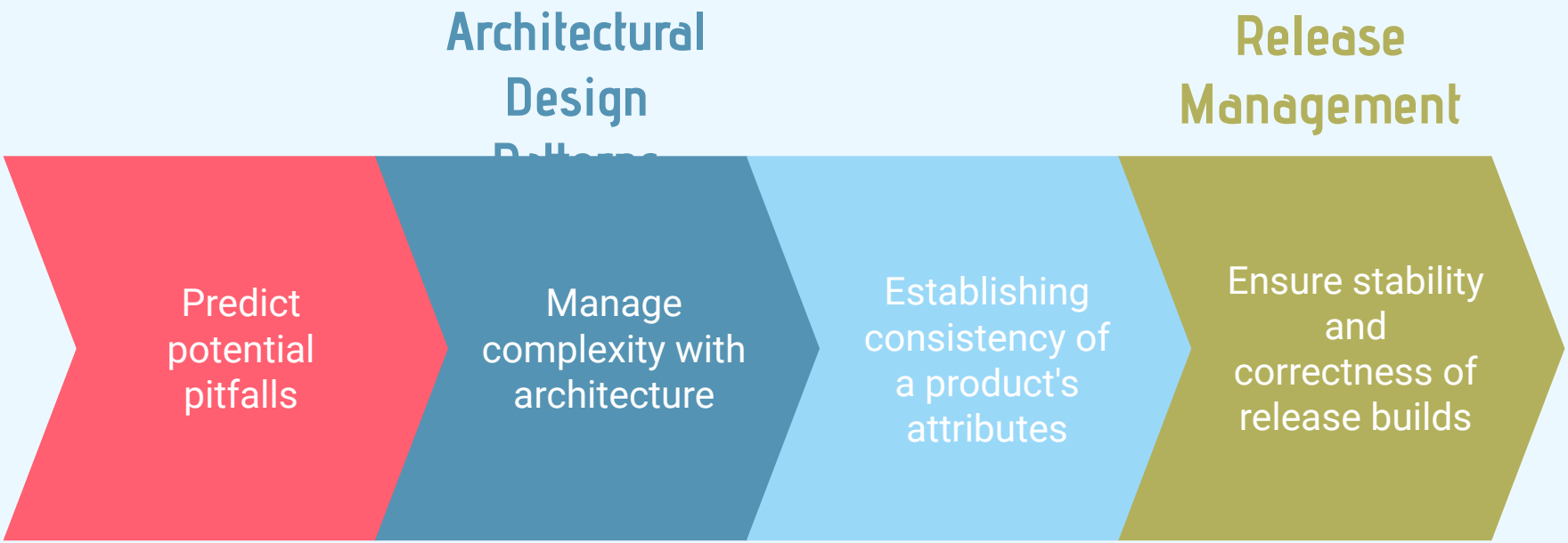
Feature #04

Users are able to access their matched-friends and selected interests history via their profile page.



02 Design for Maintainability

Approach


Architectural Design
Release Management
Predict potential pitfalls
Manage complexity with architecture
Establishing consistency of a product's attributes
Ensure stability and correctness of release builds
Early Planning
Configuration Management

Early Planning

Predict potential pitfalls

Manage complexity with architecture

Establishing consistency of a product's attributes

Ensure stability and correctness of release builds

1. Identify complex features for analysis

2. Allow room for extension

3. Standardise code with style guides

2.16.1 Definition

A nested Python function can refer to variables defined in enclosing functions, but cannot assign to them. Variable bindings are resolved using lexical scoping, that is, based on the static program text. Any assignment to a name in a block will cause Python to treat all references to that name as a local variable, even if the use precedes the assignment. If a global declaration occurs, the name is treated as a global variable.

An example of the use of this feature is:

```
def get_adder(summand1: float) -> Callable[[float], float]:
    """Returns a function that adds numbers to a given number."""
    def adder(summand2: float) -> float:
        return summand1 + summand2

    return adder
```

2.16.2 Pros

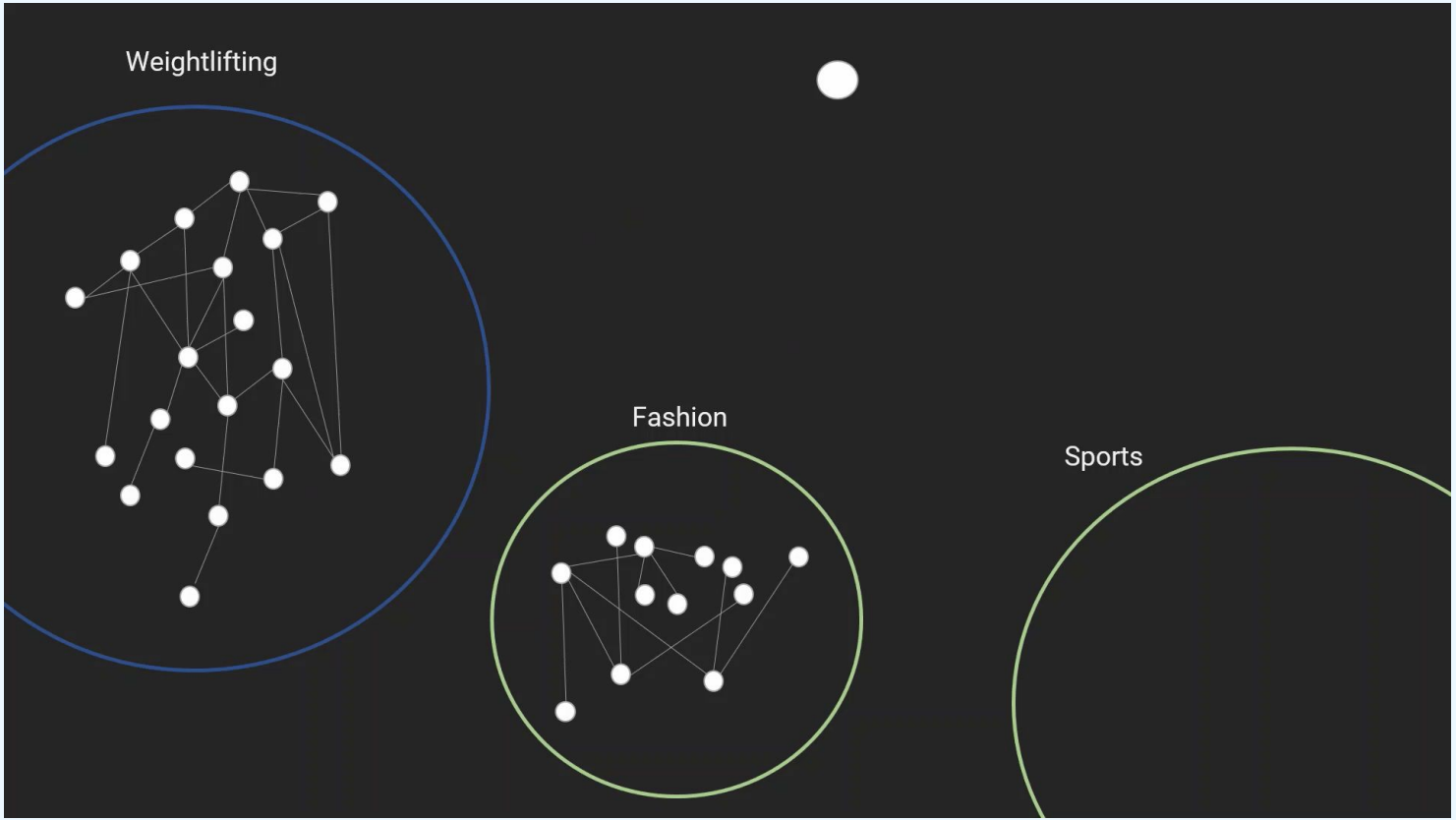
Often results in clearer, more elegant code. Especially comforting to experienced Lisp and Scheme (and Haskell and ML and ...) programmers.

2.16.3 Cons

Can lead to confusing bugs. Such as this example based on PEP-0227:

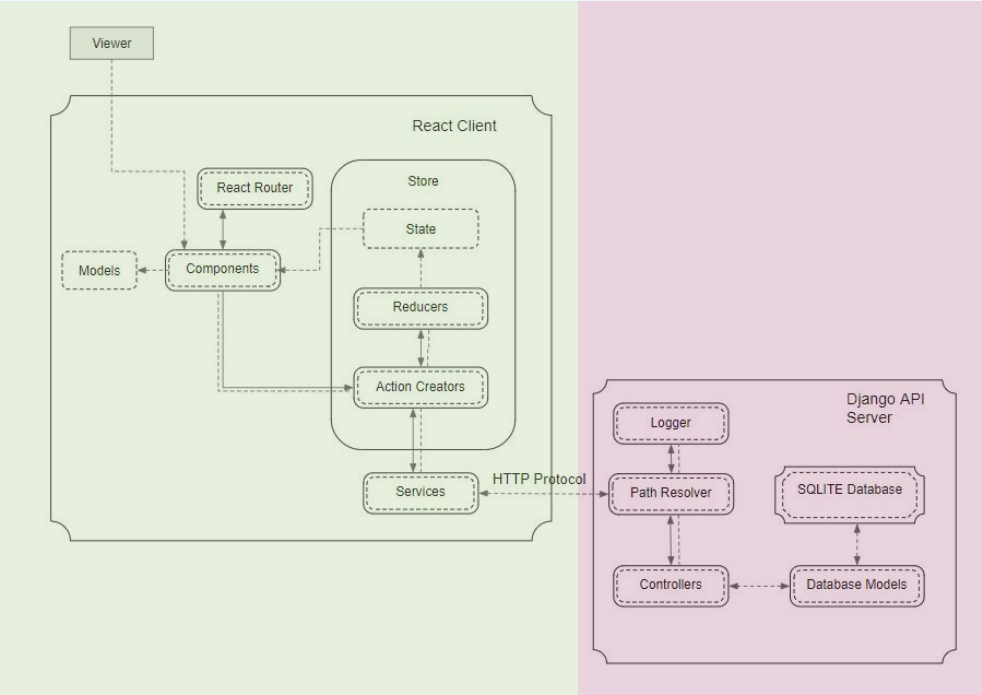
Google styleguide | Python

Complex Feature: Matching Algo



Architectural Design Patterns

Predict potential pitfalls → **Manage complexity with architecture** → Establishing consistency of a product's attributes → Ensure stability and correctness of release builds



Overall Architecture

Client-Server model

Frontend

Component-based software engineering (CBSE)

Create independent components with React

Backend

Model-View-Controller (MVC)

Use Django to implement a RESTful API

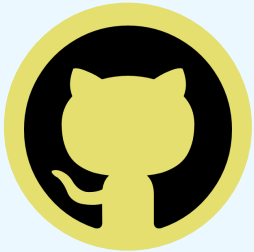
Configuration Management

Predict potential pitfalls

Manage complexity with architecture

Establishing consistency of a product's attributes

Ensure stability and correctness of release builds

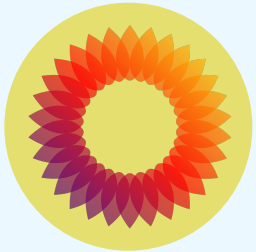


GitHub

Version Control and Source Management

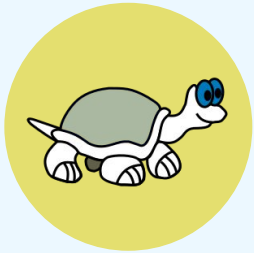
Google Drive

Documentation Collaboration



MediaWiki

Online form of Documentation



TortoiseSVN

Version Control & Documentation Management

Release Management

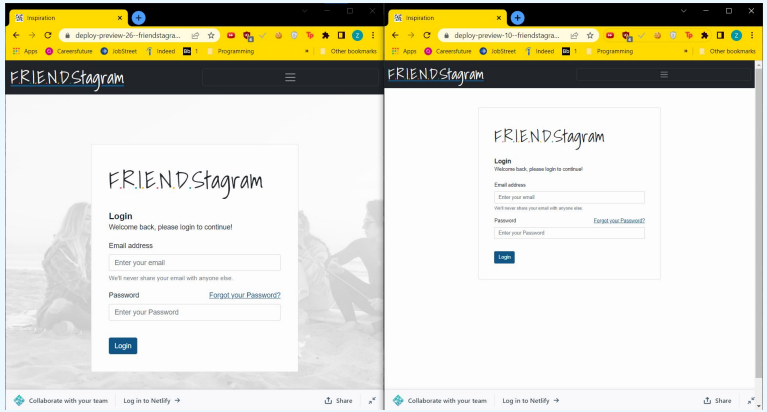
Predict potential pitfalls

Manage complexity with architecture

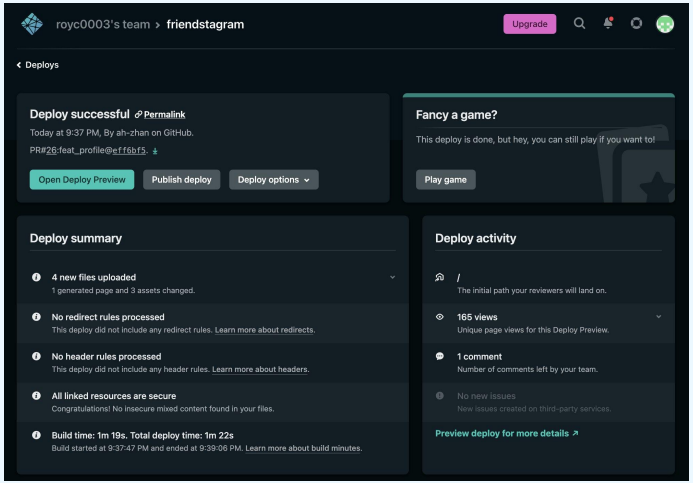
Establishing consistency of a product's attributes

Ensure stability and correctness of release builds

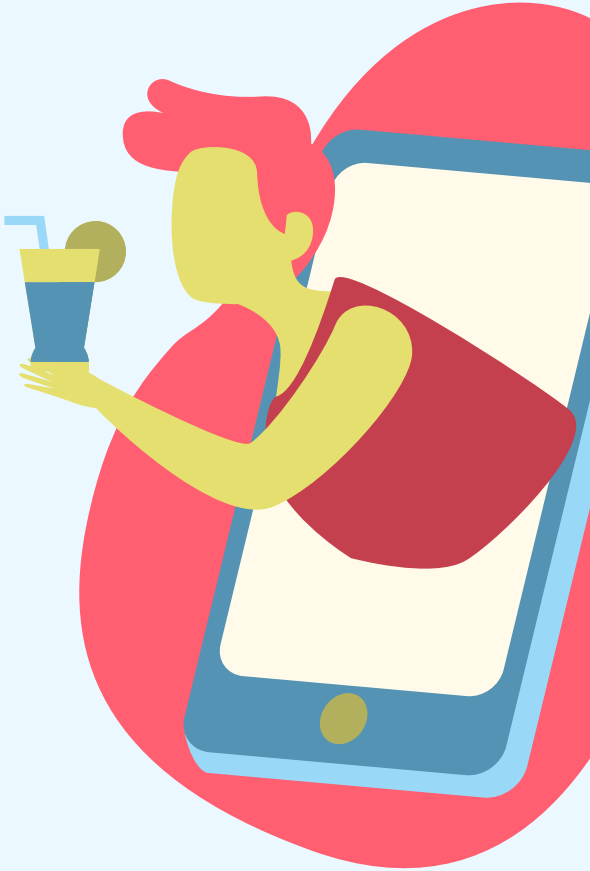
Enable stable releases via Netlify's Deploy Previews



Run automated tests on deploy builds



03 Software Quality Assurance



Key Process Areas

Process Goal

Provides the framework necessary to ensure a consistent approach to software quality assurance throughout the project life cycle

Commitment to Perform

Monitor testing efforts to assure that test schedules are adhered to and maintained to reflect an accurate progression of the testing activities.

Ability to Perform

Provision of training, resources, and tools

Activities Performed

Planning, Implementation, Review, Correction

Measurements & Analysis

Metrics used in SQA include fan-in/fan-out, length of code, length of identifiers, fog index, application crash rate, cyclomatic complexity

Verify Implementation

Ensure test management processes and products are being implemented per Test Plan.

Assessments

Product Assessment

Evaluating each Component:
User Creation, Matching, Editing Profile

Process Assessment

Evaluating each Process:
Risk Management, Configuration Management, Change Management, Release and Test

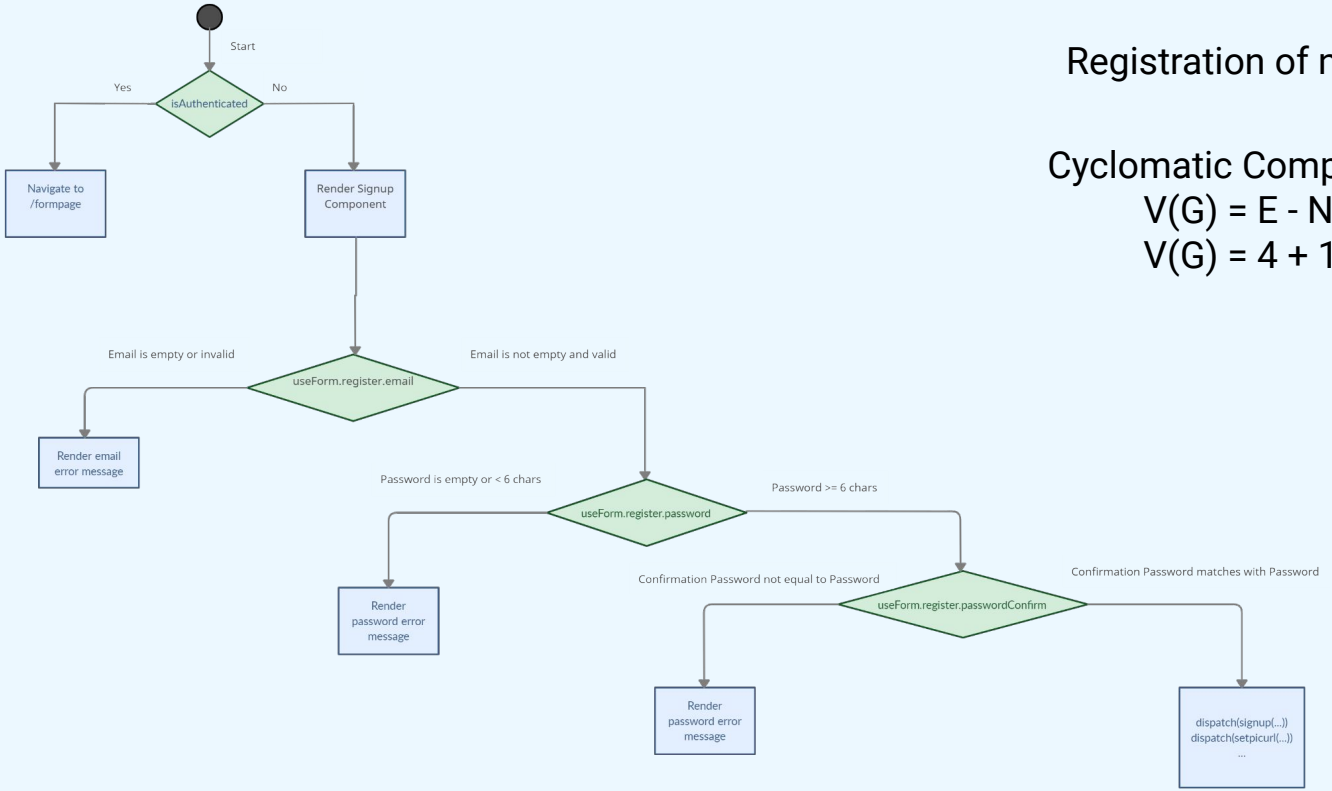
White Box Testing

Coverage report: 80%

Module	statements	missing	excluded	coverage
AccountsAndRecommendations/__init__.py	0	0	0	100%
AccountsAndRecommendations/admin.py	1	0	0	100%
AccountsAndRecommendations/apps.py	4	0	0	100%
AccountsAndRecommendations/managers.py	21	9	0	57%
AccountsAndRecommendations/migrations/0001_initial.py	8	0	0	100%
AccountsAndRecommendations/migrations/0002_auto_20220207_1656.py	7	0	0	100%
AccountsAndRecommendations/migrations/0003_auto_20220216_2053.py	4	0	0	100%
AccountsAndRecommendations/migrations/0004_auto_20220221_1659.py	4	0	0	100%
AccountsAndRecommendations/migrations/__init__.py	0	0	0	100%
AccountsAndRecommendations/models.py	132	20	0	85%
AccountsAndRecommendations/serializers.py	23	0	0	100%
AccountsAndRecommendations/tests/__init__.py	0	0	0	100%
AccountsAndRecommendations/tests/data_mockers.py	30	4	0	87%
AccountsAndRecommendations/tests/test_models.py	65	3	0	95%
AccountsAndRecommendations/tests/test_views.py	120	3	0	98%
AccountsAndRecommendations/urls.py	3	0	0	100%
AccountsAndRecommendations/views.py	183	65	0	64%
Inspiration/__init__.py	3	0	0	100%
Inspiration/asgi.py	4	4	0	0%
Inspiration/settings.py	47	5	0	89%
Inspiration/urls.py	7	0	0	100%
Inspiration/wsgi.py	4	4	0	0%
manage.py	12	2	0	83%
populate_db.py	20	20	0	0%
Total	**702**	**139**	**0**	**80%**

coverage.py v6.3.2, created at 2022-03-09 13:57 +0000

White-box Testing



Registration of new user

Cyclomatic Complexity:

$V(G) = E - N + 2 = P + 1$

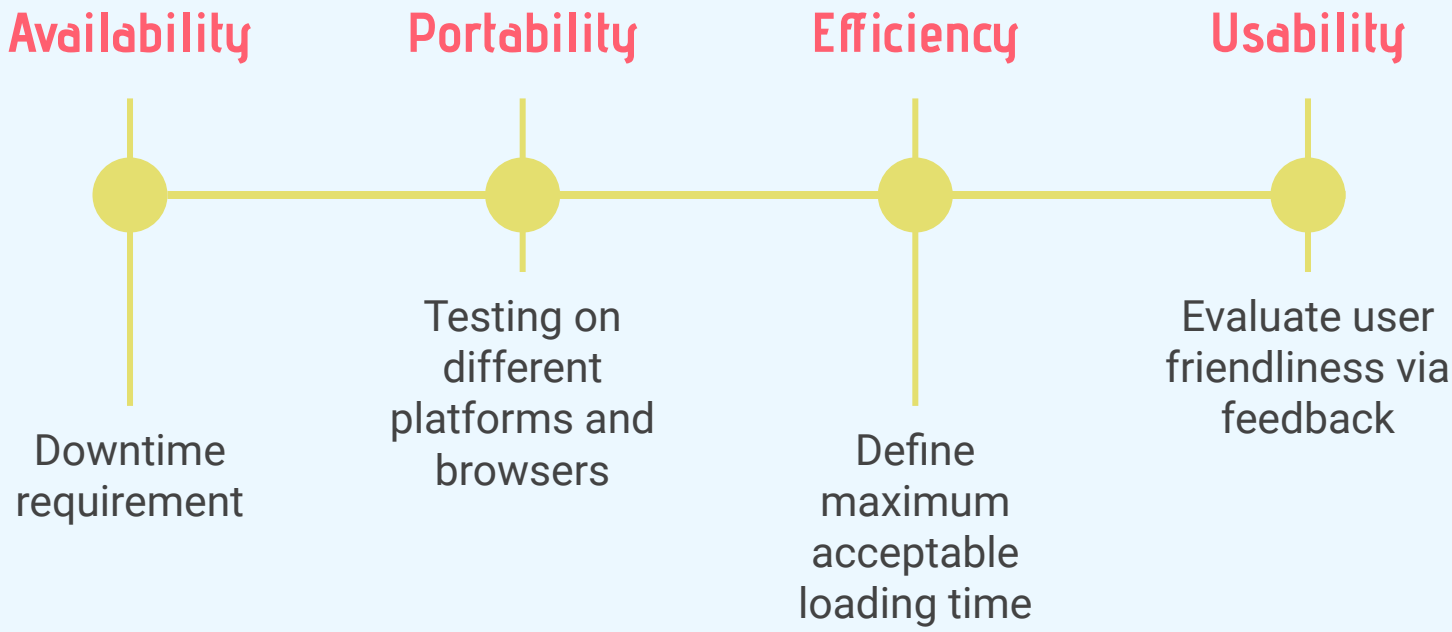
$V(G) = 4 + 1 = 5$

Black-box Testing

Questionnaire 2 - Interests Selection

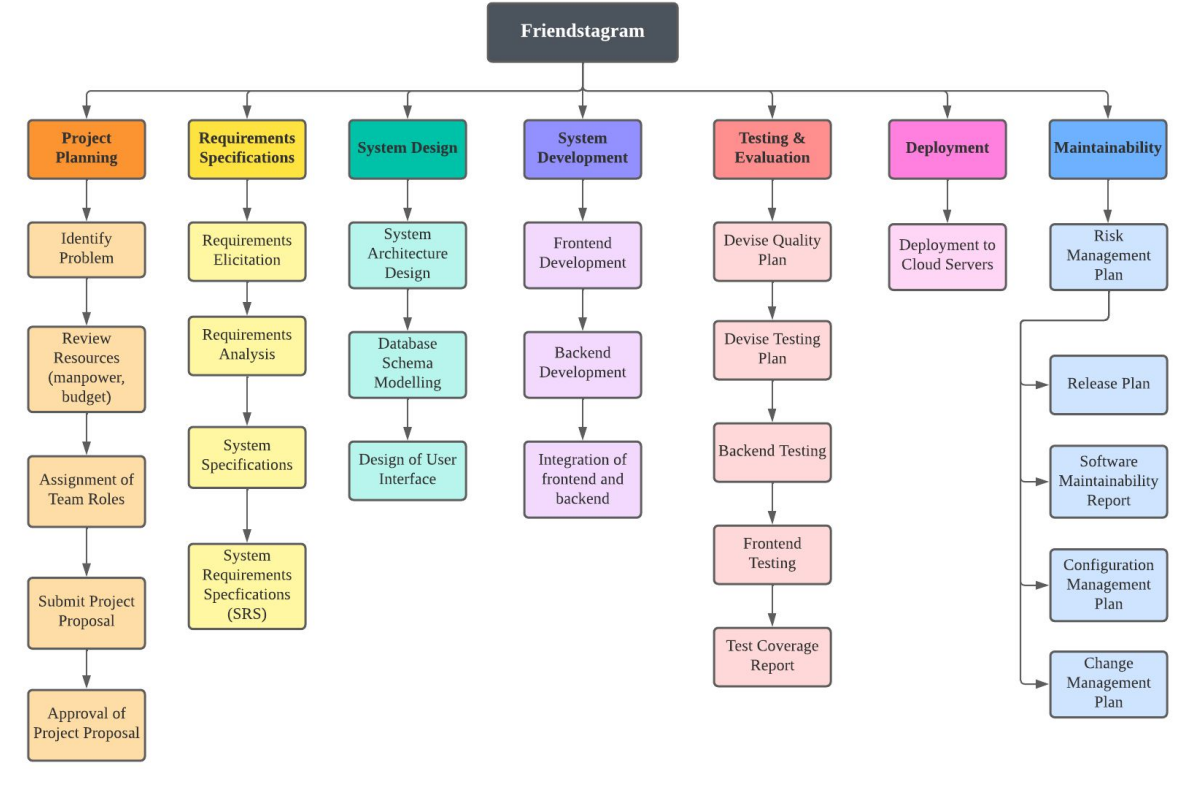
Test Cases	Input	Expected Output	Test Output
User does not select any interest.	-	"Continue to next question" button is light-blue and users are not allowed to click on it.	"Continue to next question" button is light-blue and users are not allowed to click on it.
User select an interest	playing computer games	(1) The background of the selected interest turns grey. (2) "Continue to next question" button turns dark blue and the user is allowed to click on it.	(1) The background of the selected interest turns grey. (2) "Continue to next question" button turns dark blue and the user is allowed to click on it.
User selects multiple interests	playing computer games, studying	(1) The background of the selected interest turns grey. (2) "Continue to next question" button turns dark blue and the user is allowed to click on it.	(1) The background of the selected interest turns grey. (2) "Continue to next question" button turns dark blue and the user is allowed to click on it.
User unselects interest and the total number of selection > 1	playing computer games [studying] Note: []; indicates removal	(1) When unselected, the background of the image should transit from grey to default. (2) "Continue to next question" button turns dark blue and the user is allowed to click on it.	(1) When unselected, the background of the image should transit from grey to default. (2) "Continue to next question" button turns dark blue and the user is allowed to click on it.

Software Quality



04 Project Management

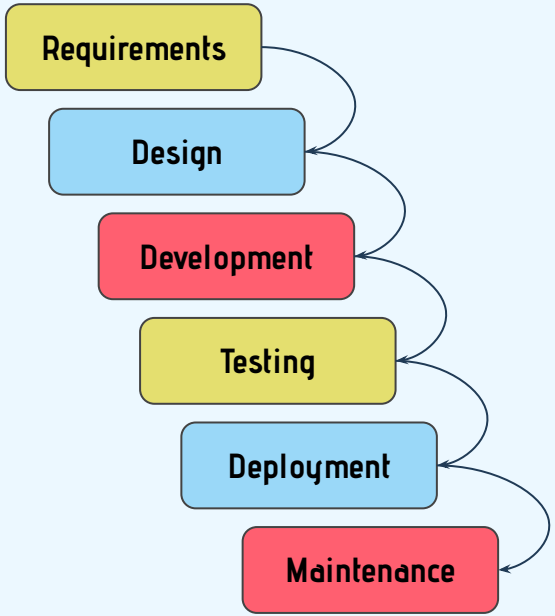
Work Breakdown Structure



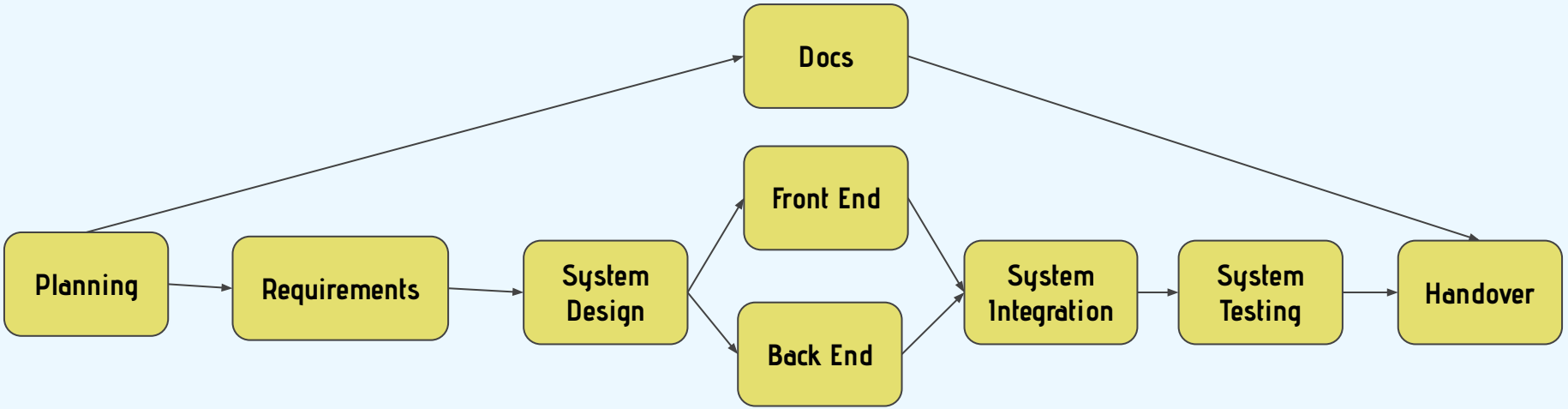
Waterfall Model

Explanation & Benefits

- Development cycle is broken down into different phases wherein each phase is connected to the next sequentially
- Each phase is well-defined and well-documented
- Allows for speedy development while meeting strict deadlines
- Clear and concise objectives across the development process.



Activity Diagram & Work Package



Best Practice Checklist

No.	Practices	✔
1	Documentation must be in a standardized format.	
2	Weekly meetings should be held in accordance with the Waterfall SDLC Model. Each member should be made aware of any changes.	
3	Changes in code must be checked by a fellow developer and must pass the specific test case, before it can be merged.	
4	Bugs and errors should be highlighted to all team members.	
5	Coding Style must be consistent and follow the preset guidelines.	
6	Project Scheduling should be done thoroughly and followed so that the project can be completed in a timely manner while meeting requirements.	
7	Software testing should be high priority and very thorough to ensure product quality and standards.	

Monitoring and Control

Measurement of Resource Consumption

Regular Reviews of Project Progress

Identification of Major Project Risks

Timeline Planning and Task Decomposition

05 Risk Management

Risks

Probability \ Impact	Low	Medium	High
High	• Lack of physical team meetings		
Med	• Internal conflict		• Requirements miscommunication • Deadline Delays
Low			• Team member unavailability

Strategies to mitigate risks

Team Member Unavailability

- Pair separation to cover
- Scope for reassignment to maintain timeline

Internal Conflict

- Ensure clear and concise communication
- Project manager should de-escalate any conflict
- Seek external help if necessary

Lack of Physical Team Meetings

- Schedule weekly online meetings
- Regular updates on team text channels

Requirements Miscommunication

- Thorough requirement solicitation and documentation
- Follow documentation closely
- Notify entire team about any change

Deadline Delays

- Good personal time management
- Schedule must take into account any constraints

Checkout our App

https://friendstagram.netlify.app/

Thank you.