

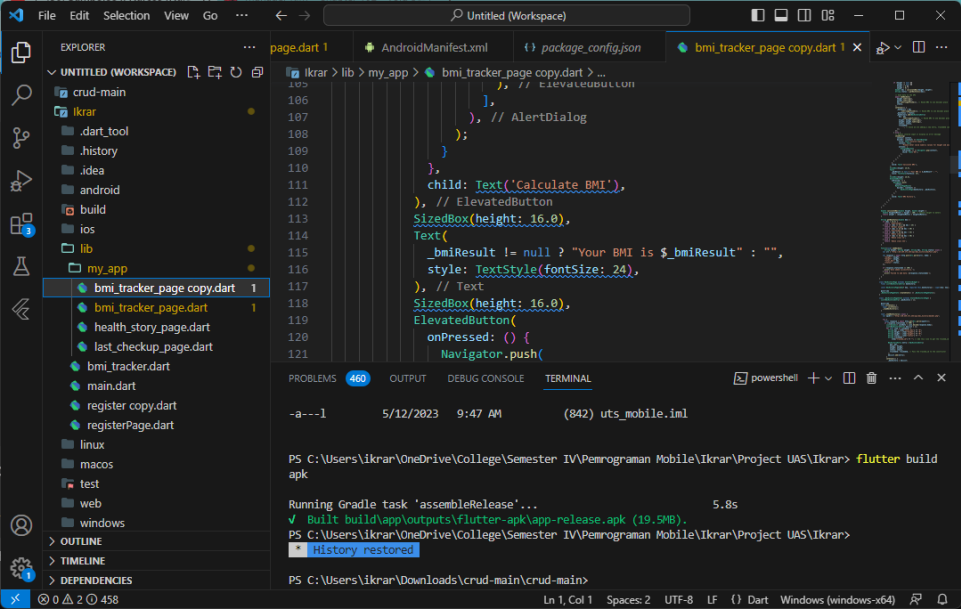
Screenshot Git

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\ikrar> git --version
git version 2.41.0.windows.1
PS C:\Users\ikrar> |
```

Screenshot Visual Studio Code



Summary

SUBJECT:	Homework Intro to Software Eng	NAME:	IKRAR BAGASKARA	↓
----------	--------------------------------	-------	-----------------	---

Fullstack Developer

Fullstack developer is a role where developers have the ability to develop systems

1. Front-end or interface of a system, where users will interact with the system. Some of the skills that a front-end developer must have are HTML, CSS, and JavaScript. As the website develops, there are also many supporting frameworks for building websites such as React, Vue.Js, Angular.Js, Stelve, Next.Js, and so on.
2. The back-end or system is responsible for processing requests from users. Programming languages such as Node.Js, Python, Ruby, and C# are used to build back-end systems. The back-end also has framework support such as Ruby on Rails, Laravel for PHP, Flash/Django for Python, and so on.
3. Database management is also an important skill that a fullstack developer must have, where the fullstack developer must understand the type of database to be used, the database model used and so on to ensure the system can run properly.

4. In addition to website development, many fullstack developers also have the ability to develop mobile applications using framework support such as Kotlin, Flutter, and React Native.

Collaboration and Version Control System

Collaboration is one of the skills that fullstack developers must have to build a good system. One of the tools supporting collaboration is using version control such as Github, GitLab, and BitBucket. The developer can choose to use version control depending on the needs of the development team. By collaborating using version control there are also advantages such as:

- History for every code change
- Version control can help identify if there are conflicts in the code base.
- Version control also provides features for code recovery to previous versions

Here are some common features available in almost all version control systems:

- Project Initialization, where all the code will be stored locally
- Parallel Development, each team member has their own copy on their local machine
- Branching, allowing teams to develop applications without interfering with the main code.
- Merge, branches that have been created for the development of new features can be merged with the main code
- Pull Request, a feature that allows developers to submit changes to their code for review by other team members.

SDLC & Design Thinking Implementation

SDLC (Software Development Life Cycle) is a structured set of processes and methods used to develop software from start to finish. The SDLC consists of a series of steps that are interrelated and performed sequentially to ensure the software development process runs smoothly and is consistent with the specified needs and objectives. SDLC also has cycles such as the following:

1. Planning and Analysis

In this stage, the team identifies the business problems and needs according to the user requirements. The plan also includes resource allocation, timelines, and team member responsibilities.

2. Design

Design the software in detail including system architecture, user interface, and database design.

3. Development

Implementation of a previously designed and agreed upon product with the goal of producing a working product.

4. Testing

Ensuring the developed product works as required, testing also covers the functionality, performance, safety, and quality of the entire product.

5. Implementation & Integration

Implementation and integration in SDLC are important steps to combine software components and ensure optimal quality and performance.

6. Maintenance

Maintain bugs as they arise, and enhance features as needed.

By using SDLC effectively, organizations can increase success and efficiency in developing applications, ensure timely delivery of quality products, and provide greater value to customers and stakeholders.

SDLC Model

- **Waterfall Model**

Sequential SDLC model, each stage must be completed before starting the next, suitable for projects with clear and stable requirements.

- **V-Shaped Model**

Has similarities to the Waterfall model but emphasizes testing, the slash "V" means the development has appropriate testing. Suitable on projects with a high quality focus

- **Prototype Model**

Aim to create an initial example before developing the final version. By focusing on user needs and gathering feedback.

- **Spiral Model**

Each cycle builds on the previous incremental, resulting in software that progressively evolves with more features, suitable for large and complex projects.

- **Iterative Incremental Model**

Iterating and improving the product in small stages. Each iteration adds more features until it is completed according to the goal, suitable for projects with limited time and budget.

- **Big Bang Model**

A model where the development stages are carried out without in-depth planning and analysis. Suitable for small projects

- **Agile Model**

The collaborative and iterative approach focuses on periodic feature development. Teams work in sprints or timelines. Suitable for projects with dynamic environments and requirements

Design Thinking Implementation

1. **Empathize: Understand User Needs**

Deeply understand user needs, problems, and desires.

Key points: User Research, Empathy Mapping, User Personas

2. **Define: Define the Problem**

Analyze the information from the empathy stage and then set the goals of the project.

Key points: Problem Statement, Stakeholder Alignment

3. **Ideate: Generate Ideas**

Brainstorming to find solutions to the problems that arose.

Key points: Brainstorming Sessions, Idea Consolidation

4. **Prototype: Build and Iterative Solutions**

Create a tangible representation of the ideas that have been selected.

Key Points: Low Fidelity, High Fidelity

5. **Test: Gather User Feedback**

Collect feedback from users.

Key points: Usability Testing, Iterative Testing

6. **Implement: Develop the Software**

Build the software based from the design that has been made.

Key points: Agile Development, Cross-Functional Collaboration

Basic Git & Collaborating Using Git

Version Control Git

Version control is a method used to track and manage changes in source code or project files. And Git is one of the most popular and powerful distributed version control systems.

- **Centralized Version Control System**

With a centralized system, one repository acts as the '**master**' for storing projects. Each developer then has a local copy of the project, and submits changes to the '**master**' repo.

- **Distributed Version Control System)**

Each member of the development team has a complete copy of the entire repository. And no one has a 'master' repository system.