

CASE Tool for Automation of Structured Software Analysis and Design

## PROJECT REPORT

Submitted for the Course

CSE3001-Software Engineering

by

### Saksham Dewan (19BCE2090)

### Aayush Gupta (19BCE0530)

### Akshat Bakliwal (18BCE0787)

### 

Slot: D1

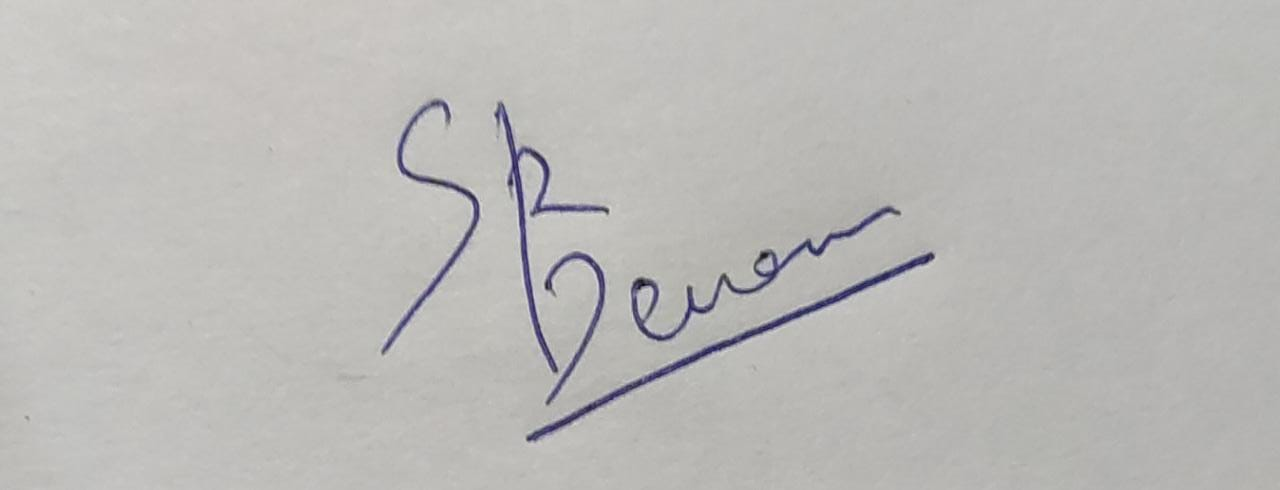
Name of faculty: Dr. SWATHI J.N

## SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

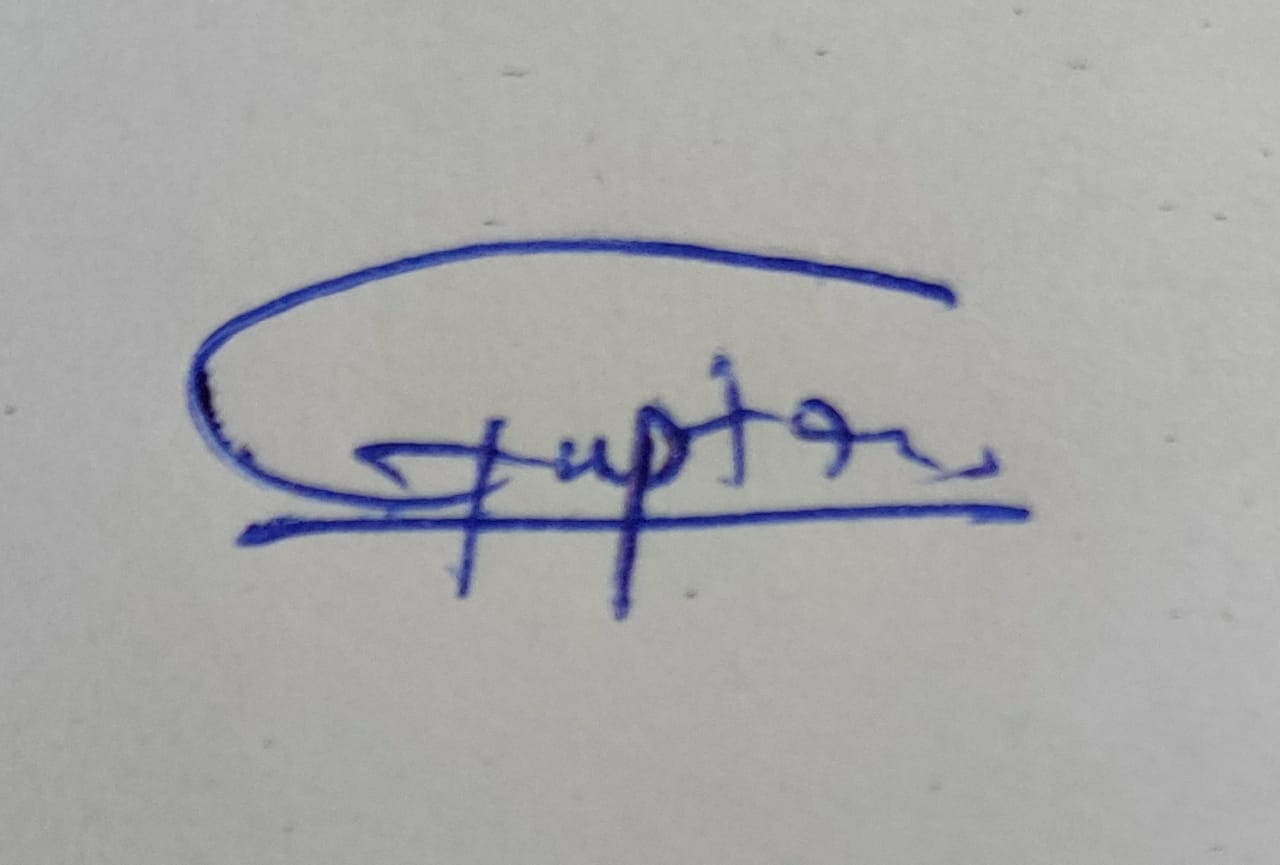
**ACKNOWLEDGEMENTS**

### We take immense pleasure in thanking Dr. G. Viswanathan, our respected Chancellor, VIT University and respected Dean, Dr. R. Saravanan, for having permitted us to carry out the project. We express gratitude to our guide and subject teacher, Dr. Swathi J.N, for guidance and suggestions that helped us to complete the project on time. She corrected our mistakes and guided us to build better software by giving us better suggestions .

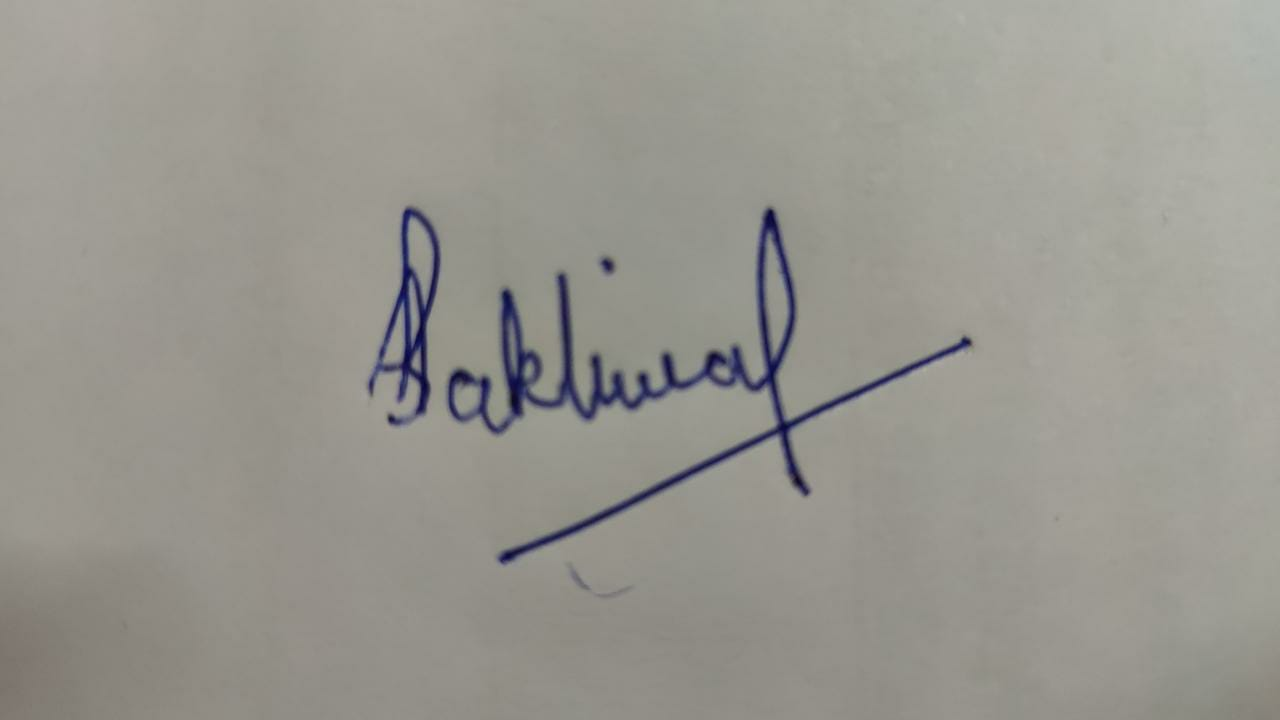
Signature of Student 1



Signature of Student 2



Signature of Student 3



# Executive Summary

The CASE tool for Automation of structured software analysis and design plays an important role in the active research areas of software engineering ,teaching and understanding DFDs.Data flow diagrams are used to graphically represent the flow of data in a business information system. DFD describes the processes that are involved in a system to transfer data from the input to the file storage and reports generation.DFD graphically represents the functions, or processes, which capture, manipulate, store, and distribute data between a system and its environment and between components of a system. The visual representation makes it a good communication tool between User and System designer. Structure of DFD allows starting from a broad overview and expands it to a hierarchy of detailed diagrams. Thus this tool creates effective DFDs which provide a lot of information about the structured analysis and the design for any product manufacturing or any service .

*Keywords: Data Flow Diagram (DFD), CASE (Computer-Aided Software Engineering), Software Engineering*

# Table of Contents

**CONTENT Page**

**Acknowledgement 2**

[**Executive Summary 3**](#_gjdgxs)

[**Table of Contents 4**](#_30j0zll)

**INTRODUCTION 5**

* 1. [Objective 5](#_1fob9te)
  2. [Motivation 5](#_3znysh7)
  3. [Background study 5](#_2et92p0)

1. **PROJECT DESCRIPTION AND GOALS 6**
2. [**TECHNICAL SPECIFICATION 7**](#_tyjcwt)
3. **DESIGN AND APPROACH DETAILS 8**
   1. Analysis and Design 8
   2. [CODE 10](#_3dy6vkm)
4. **SCHEDULE, TASKS & MILESTONES 14**
5. [**PROJECT DEMONSTRATION & RESULTS 17**](#_1t3h5sf)
6. [**CONCLUSION 18**](#_4d34og8)
7. [**REFERENCES 18**](#_2s8eyo1)
8. **INTRODUCTION**

## OBJECTIVE

The projects aims to achieve the following-

* Tool should support a graphical interface and the following features.
* The user should be able to draw bubbles, data stores, and entities and connect them using data flow arrows. The data flow arrows are annotated by the corresponding data names.
* Should support editing the data flow diagram.
* Should be able to create the diagram hierarchically.
* The user should be able to determine balancing errors whenever required.
* The software should be able to create the data dictionary automatically.
* Should support printing the diagram on a variety of printers.

## MOTIVATION

As we know that DFD is used to represent the data flow in the model which is a very essential aspect of any software that was the motivation for us to build a CASE tool on which DFD diagrams can be made. The DFD notation is based on graph theory, which was first used in operational research to model organisational processes. At the end of the 1970s, the Activity Diagram was utilised in the SADT (Structured Analysis and Design Technique) methodology. Edward Yourdon, Larry Constantine, Tom DeMarco, Chris Gane, and Trish Sarson are among the DFD promoters. Although they might theoretically be used for business process modelling, DFDs were mostly employed to depict data flow in a computer system. DFDs were beneficial for documenting key data flows or experimenting with a new high-level data flow design. The DFD also includes information on each entity's outputs and inputs, as well as the process itself. There are no control flows, decision rules, or loops in a data-flow diagram. A flowchart can be used to represent specific operations based on data.

## BACKGROUND STUDY

A data-flow diagram is a visual representation of data flowing through a system or a process (usually an information system). The DFD also includes information on each entity's outputs and inputs, as well as the process itself. There are no control flows, decision rules, or loops in a data-flow diagram. A flowchart can be used to represent specific operations based on data.

Data-flow diagrams can be displayed using a variety of notations. Tom DeMarco described the notation given above as part of Structured Analysis in 1979.

A process must have at least one of the endpoints (source and/or destination) for each data flow. Another data-flow diagram can be used to refine the representation of a process by subdividing it into sub-processes.

The structured-analysis modelling tools include a data-flow diagram. When working with UML, the activity diagram usually replaces the data-flow diagram. A site-oriented data-flow plan is a type of data-flow strategy.

Because sites in such networks correlate to the semantics of data memory, data-flow diagrams can be thought of as inverted Petri nets. Similarly, the semantics of Petri net transitions and data flows and functions from data-flow diagrams should be regarded as comparable.

DFD consists of processes, flows, entities, and data stores. There are several ways to view these DFD components.

Process:

The process (function, transformation) is part of a system that transforms inputs to outputs. The symbol of a process is a circle, an oval, a rectangle or a rectangle with rounded corners (according to the type of notation). The process is named in one word, a short sentence, or a phrase that is clearly to express its essence.

Data Flow:

Data flow (flow, dataflow) shows the transfer of information (sometimes also material) from one part of the system to another. The symbol of the flow is the arrow. The flow should have a name that determines what information (or what material) is being moved. Exceptions are flows where it is clear what information is transferred through the entities that are linked to these flows. Material shifts are modeled in systems that are not merely informative. Flow should only transmit one type of information (material). The arrow shows the flow direction (it can also be bi-directional if the information to/from the entity is logically dependent - e.g. question and answer). Flows link processes, data stores and entities.

Warehouse:

The warehouse (datastore, data store, file, database) is used to store data for later use. The symbol of the store is two horizontal lines, the other way of view is shown in the DFD Notation. The name of the warehouse is a plural noun (e.g. orders) - it derives from the input and output streams of the warehouse. The warehouse does not have to be just a data file, for example, a folder with documents, a filing cabinet, and optical discs. Therefore, viewing the warehouse in DFD is independent of implementation. The flow from the warehouse usually represents the reading of the data stored in the warehouse, and the flow to the warehouse usually expresses data entry or updating (sometimes also deleting data). Warehouse is represented by two parallel lines between which the memory name is located (it can be modeled as a UML buffer node).

Terminator:

The Terminator is an external entity that communicates with the system and stands outside of the system. It can be, for example, various organizations (eg a bank), groups of people (e.g. customers), authorities (e.g. a tax office) or a department (e.g. a human-resources department) of the same organization, which does not belong to the model system. The terminator may be another system with which the modeled system communicates.

## PROJECT DESCRIPTION AND GOALS

The goal of this project is to ease the process of structuring and analysis of activities or some project by making simple and DFDs (Data Flow Diagrams) .Data flow diagrams allow you to take any flow of information for a process or system and condense it into a coherent, easily understood visual. Using standardized circles, arrows, and text labels, you can map out entire processes and systems for even the most non-technical employee to understand. With customizable files, well-stocked DFD shape libraries, and easy-to-use formatting tools, our data flow diagram software will help you to create a simple, complex, or even multi-layered DFD

## TECHNICAL SPECIFICATION

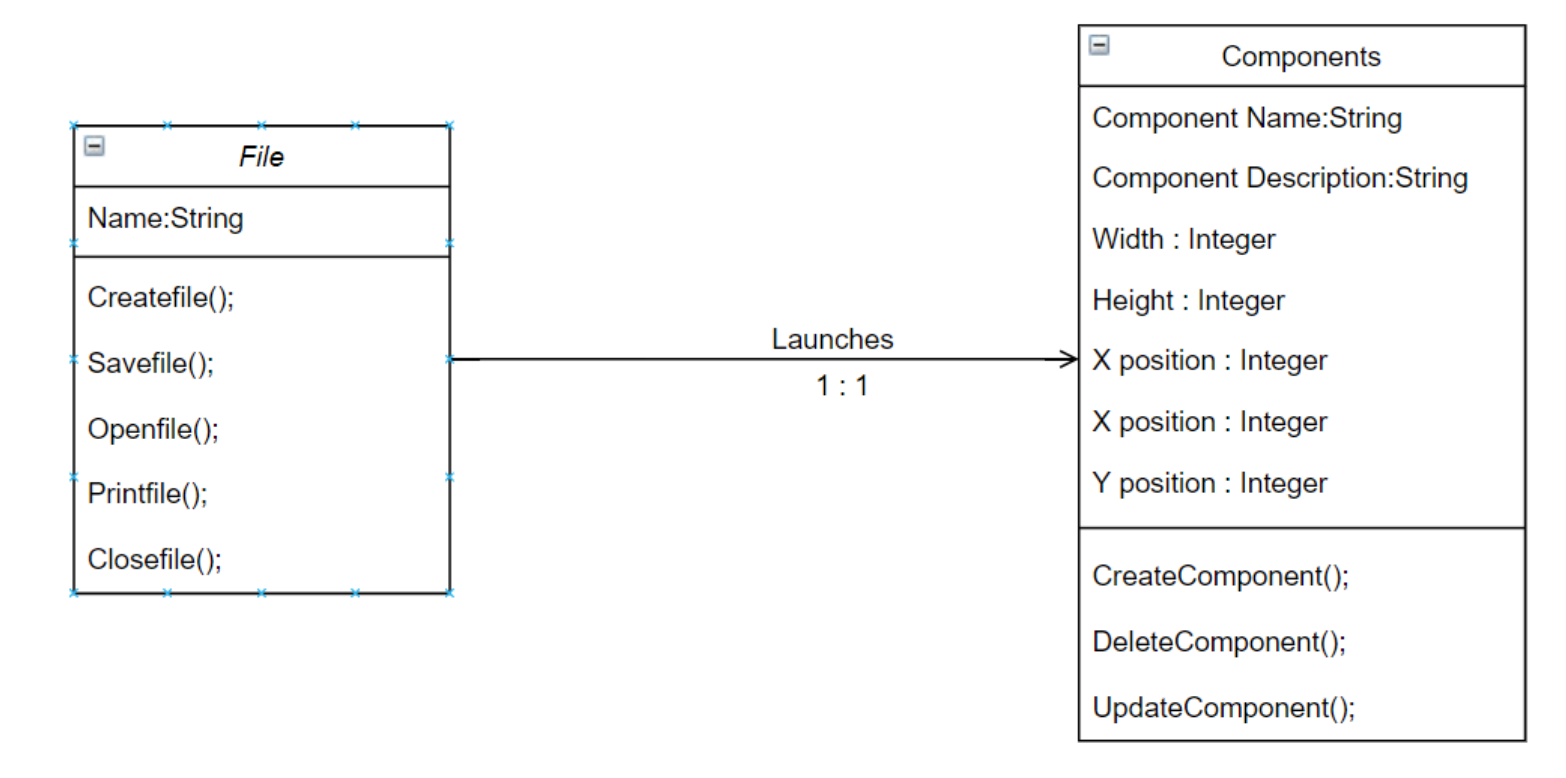
**>> Java:** Java is an object-oriented programming language with a high level of abstraction and as few implementation dependencies as possible. It is a general-purpose programming language designed to allow application developers to write once and run anywhere (WORA), which means that compiled Java code can run on any platform that supports Java without requiring recompilation. Java apps are a type of software that runs on a computer.

**>> Apache Netbeans:** NetBeans is a Java-based integrated development environment (IDE). NetBeans enables the creation of applications from a set of modular software components known as modules. NetBeans is compatible with Windows, Mac OS X, Linux, and Solaris. It has extensions for PHP, C, C++, HTML5,[3] and JavaScript, in addition to Java development.

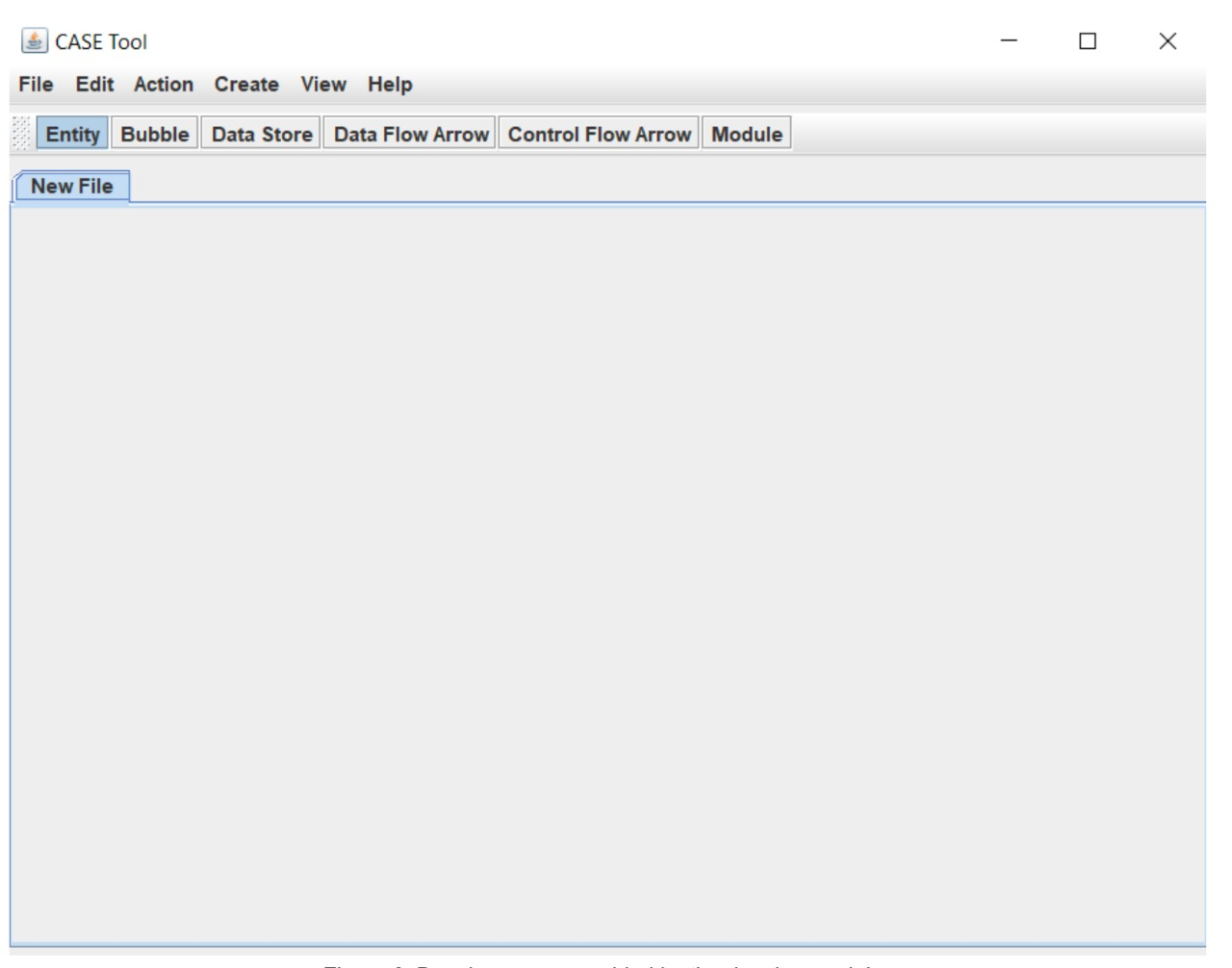
**>> JDK 15 (Java Development Kit) :** The Java Development Kit (JDK) is a binary package that implements one of Oracle Corporation's Java Platform, Standard Edition, Java Platform, Enterprise Edition, or Java Platform, Micro Edition platforms for Java developers running Solaris, Linux, macOS, or Windows.

# DESIGN & APPROACH DETAILS

## 4.1. Analysis & Design

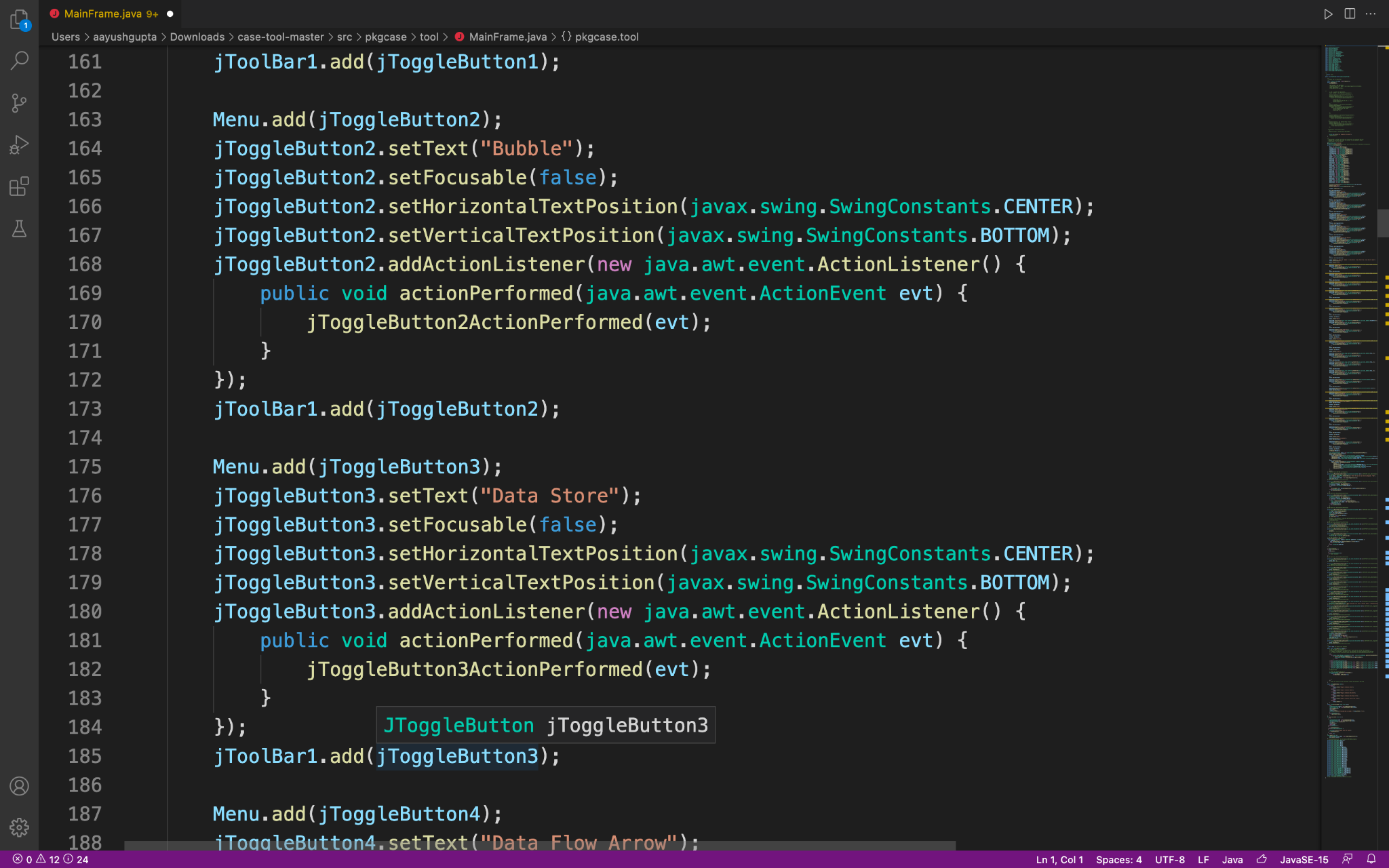


## The front module gives the area to create your DFD’s and the workflow area. We have opted for the classic waterfall model because requirements are clear and consistent beforehand. Software is light and not much heavy. Project is short and the definition is clear. Environment is stable. Technology and tools used are stable and not dynamic. Ample resources with required expertise are available to support the product.

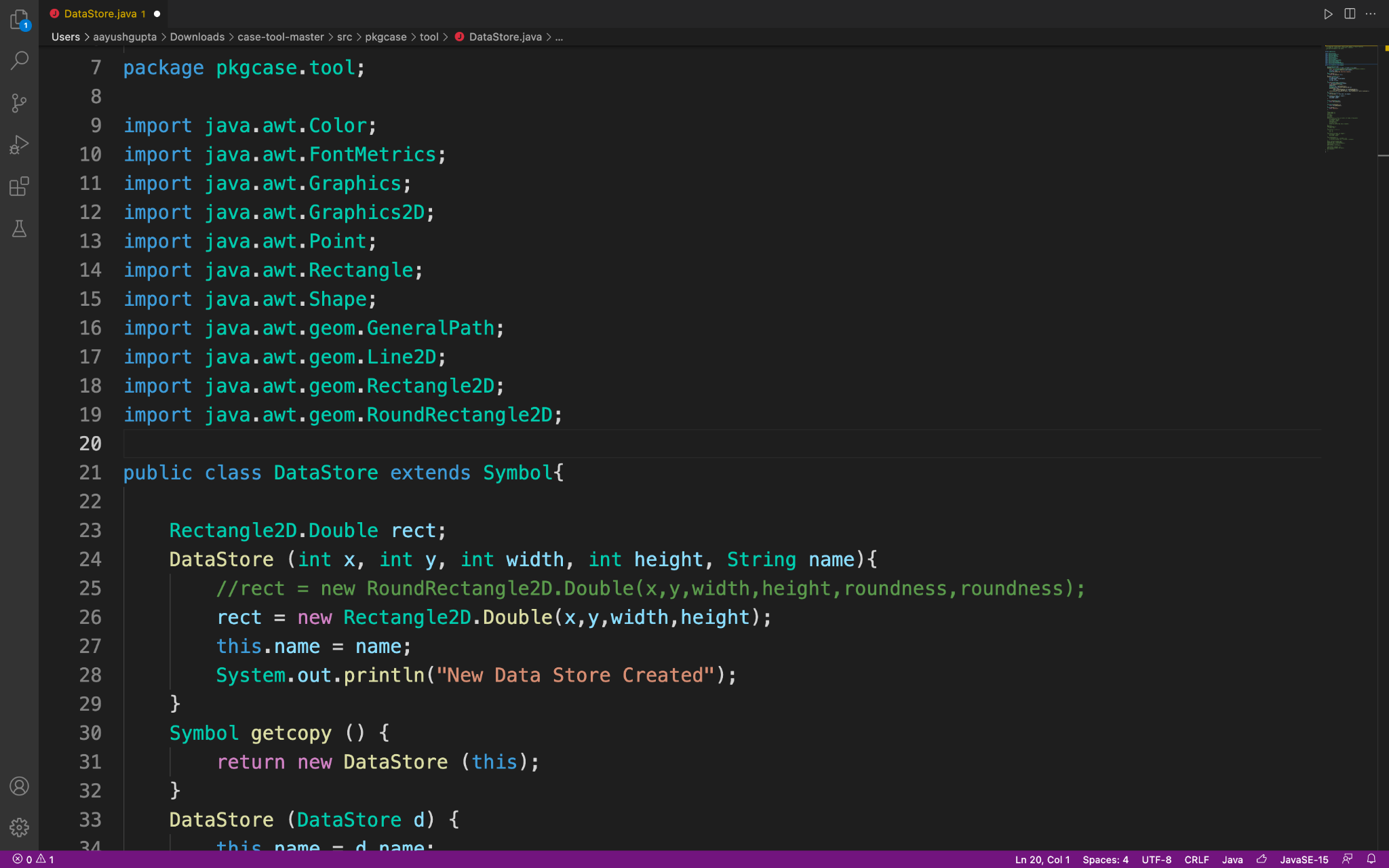


# 4.2 CODE:

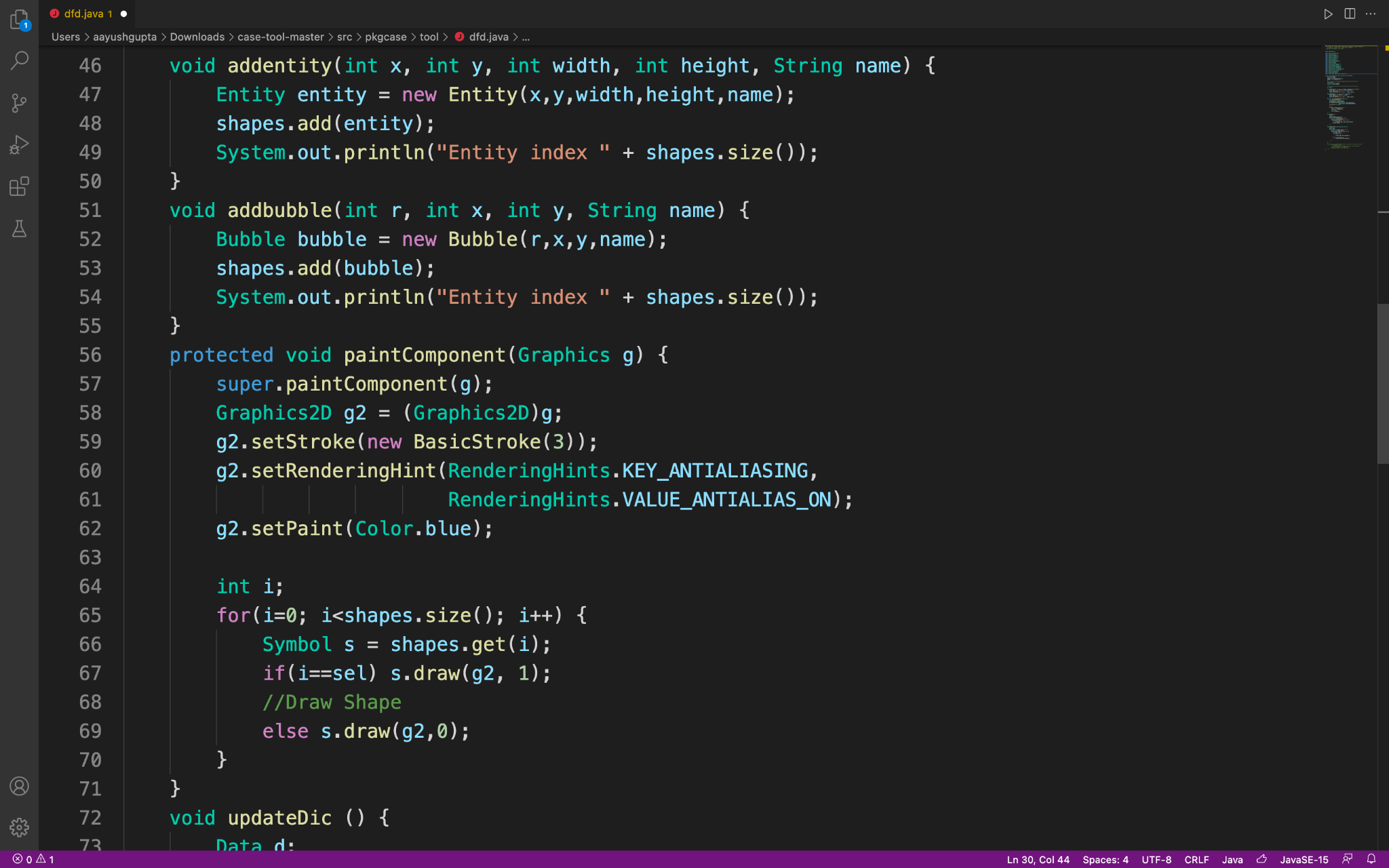
**>> MainFrame.java:**

****

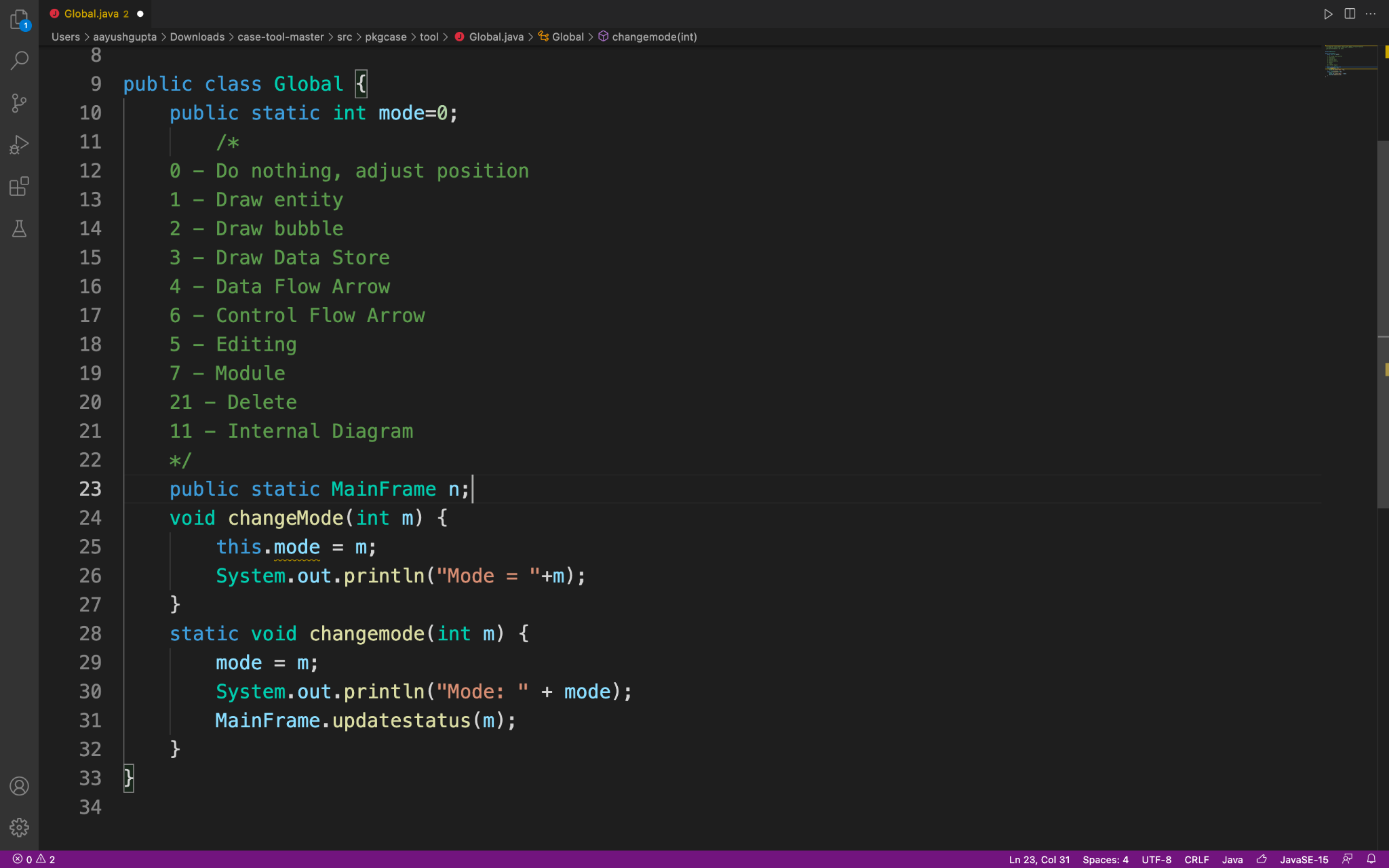
**>> DataStore.java:**

****

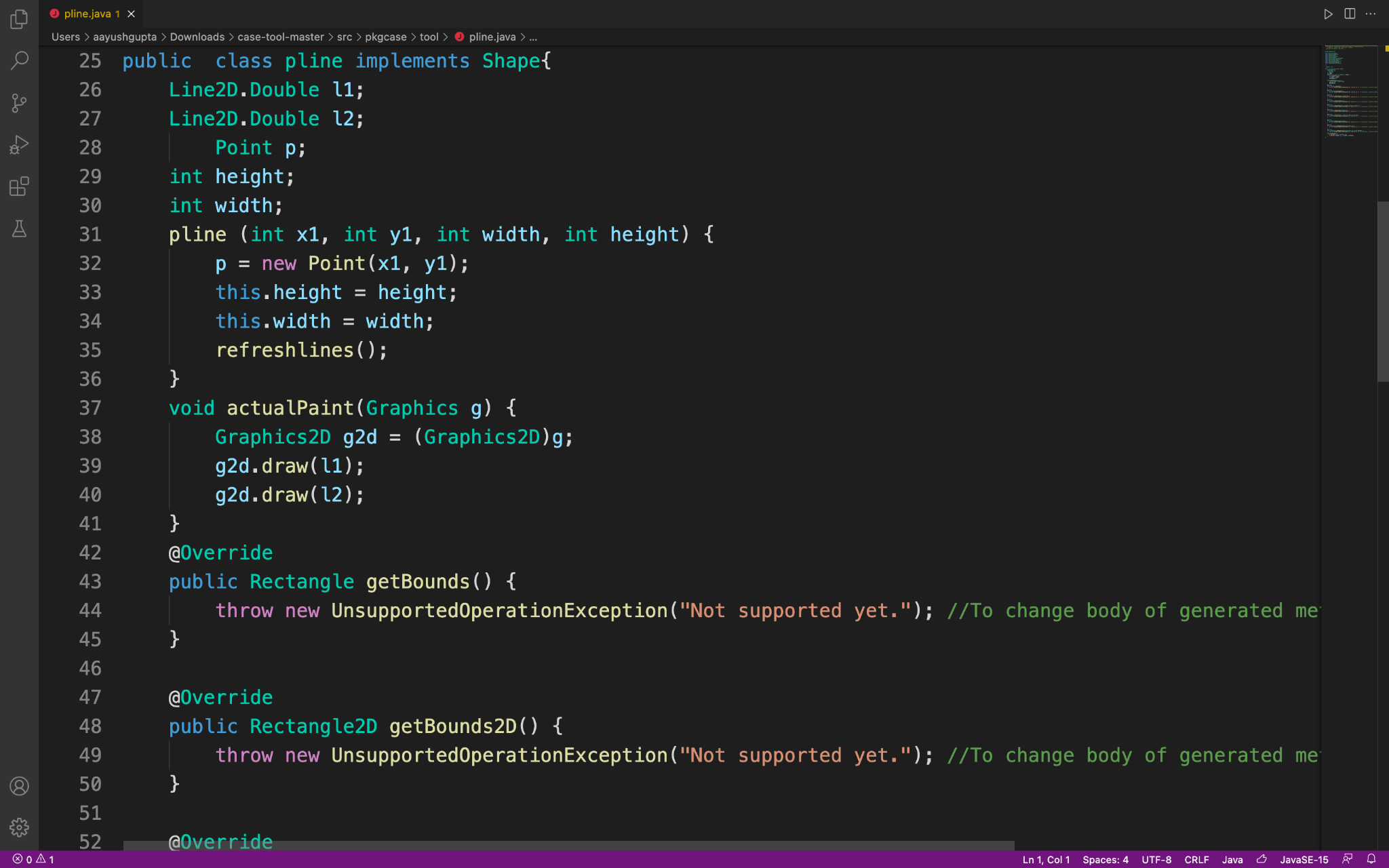
**>> dfd.java:**

****

**>> Global.java:**

****

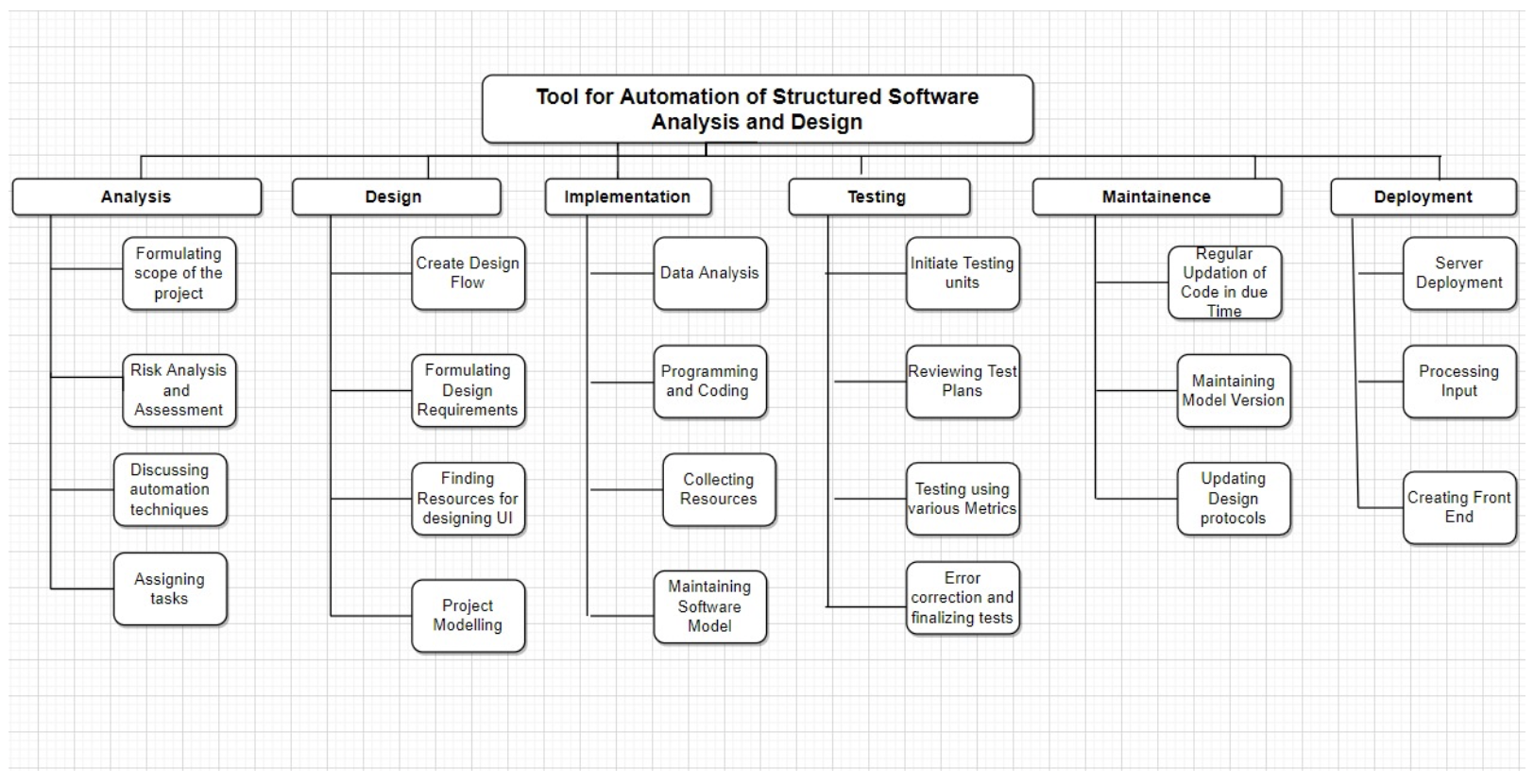
**>> pline.java:**

****

1. **SCHEDULE, TASKS AND MILESTONES**

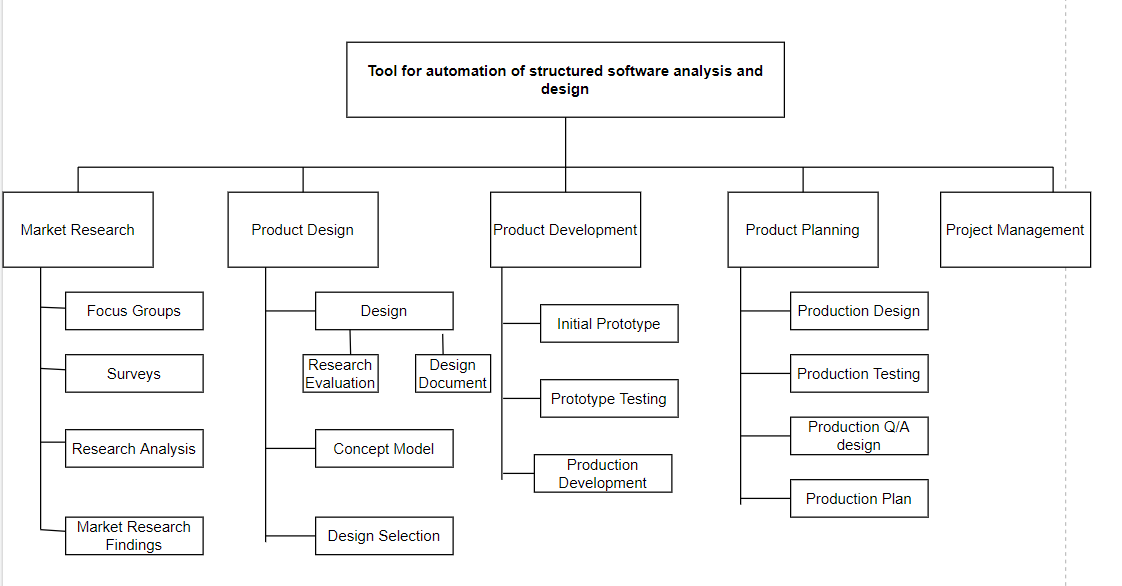
**WORK BREAKDOWN STRUCTURE FOR PROCESS**

We have used the Draw.io open-source tool to draw the WBS for the process.



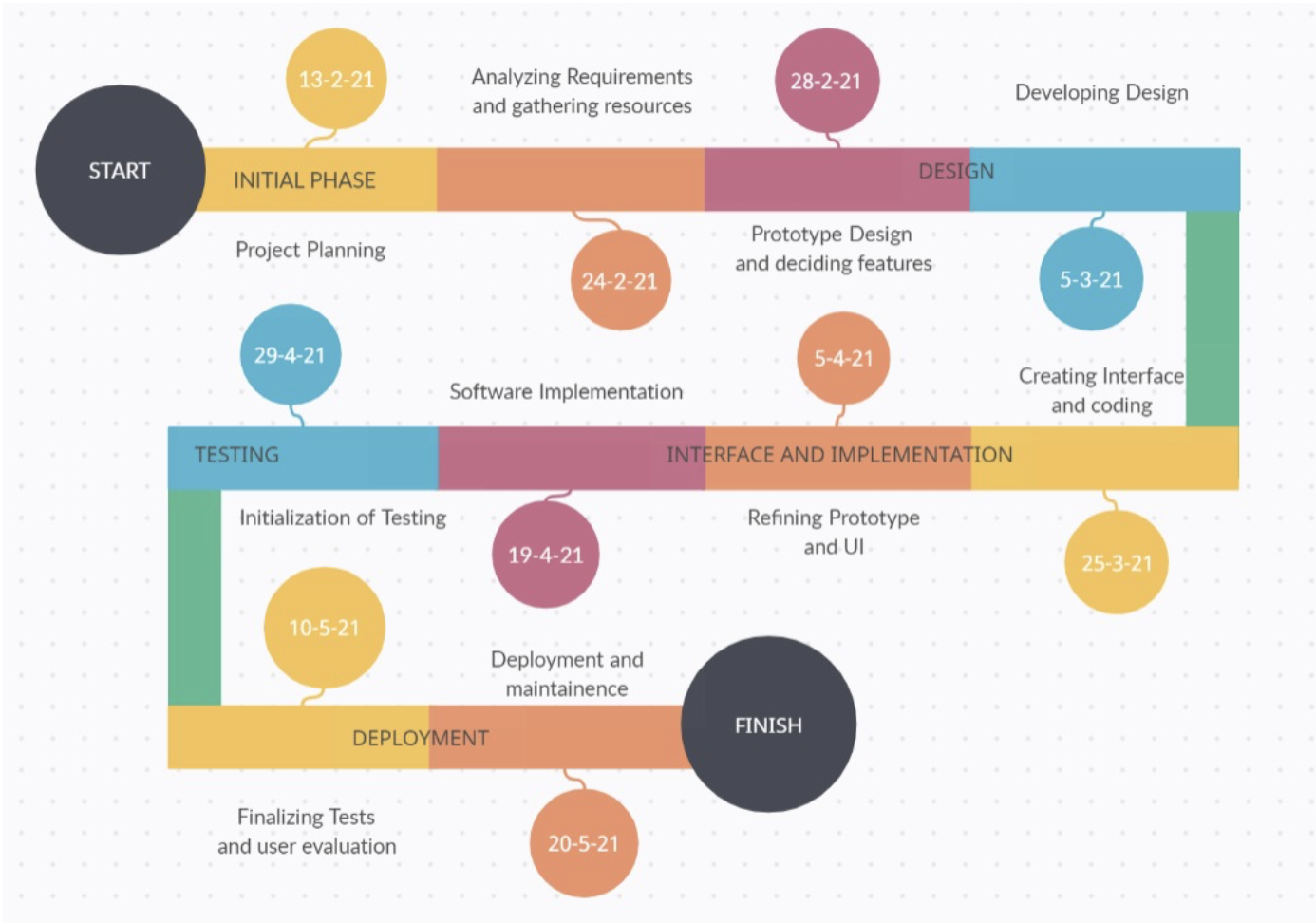
**WORK BREAKDOWN STRUCTURE FOR PROCESS**

We have used the Draw.io open-source tool to draw the WBS for the product.



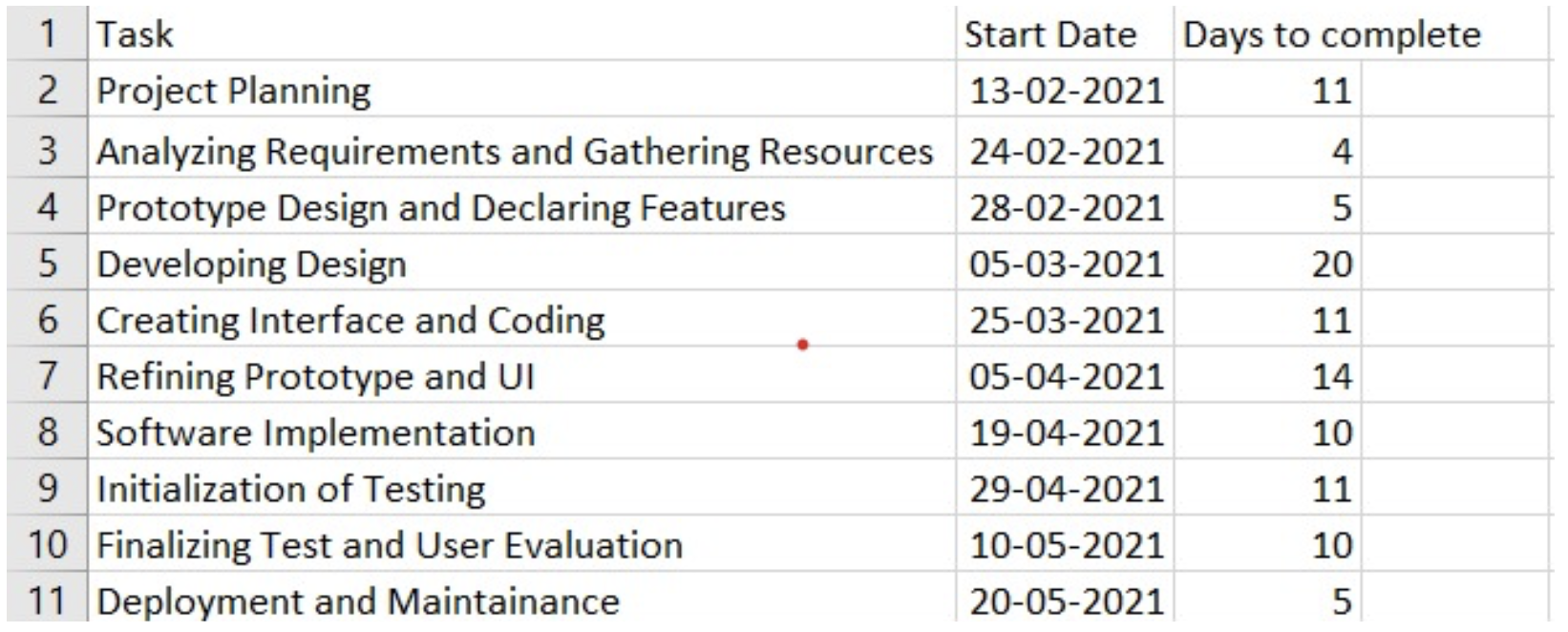
**TIMELINE CHART**

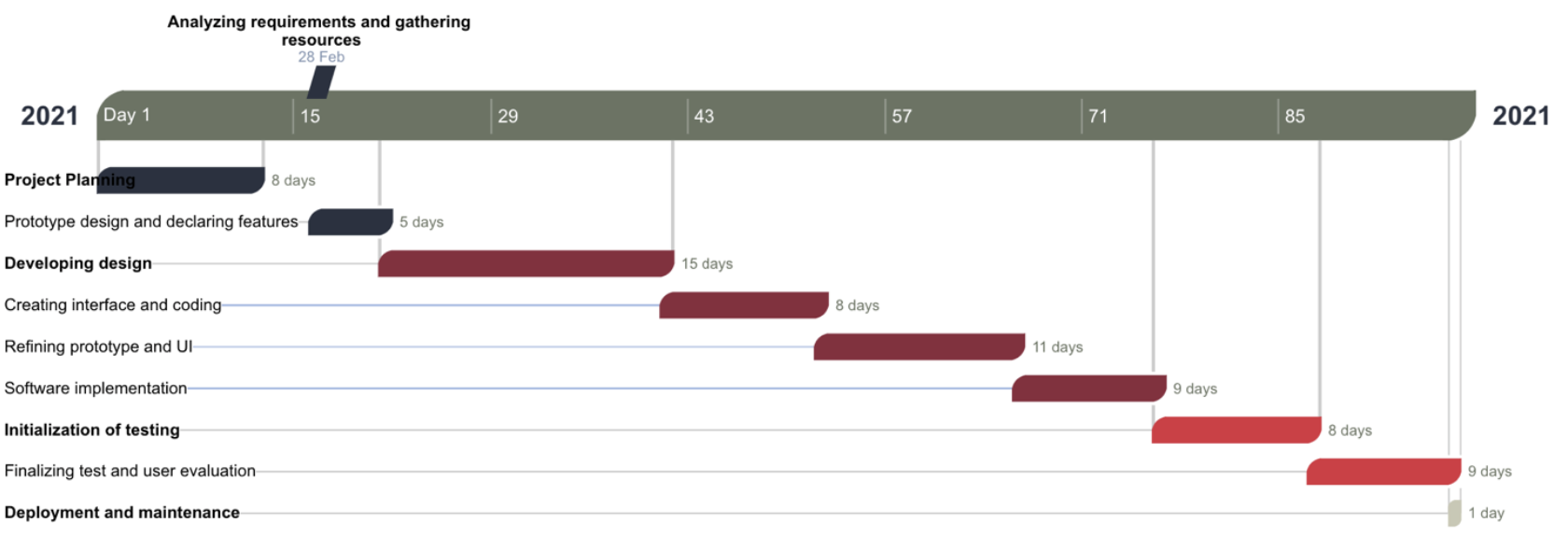
We have used the Creately open-source tool to draw the Gantt chart.



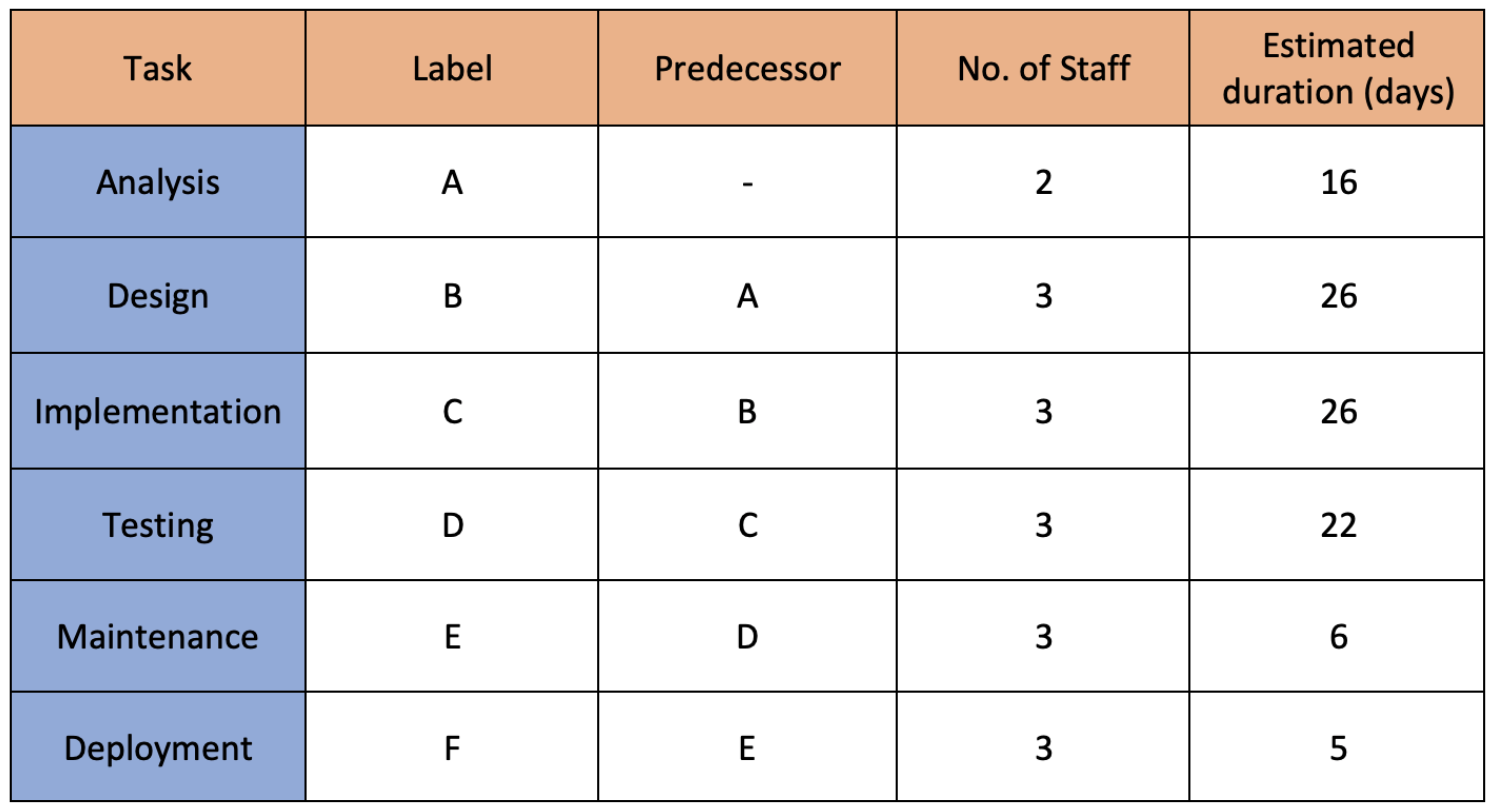
**GANTT CHART**

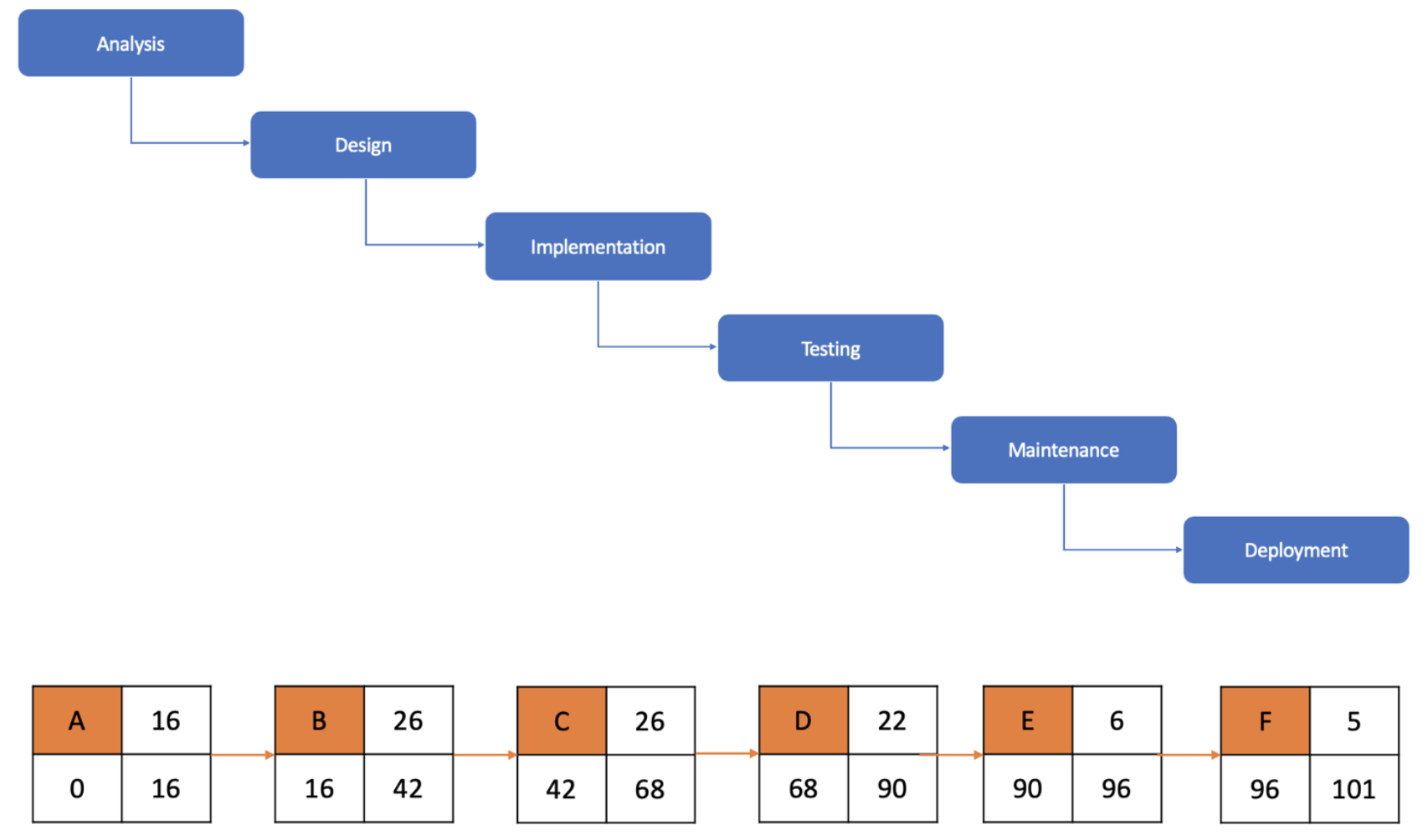
We have used Lucid Charts open-source tool to draw the Gantt chart.



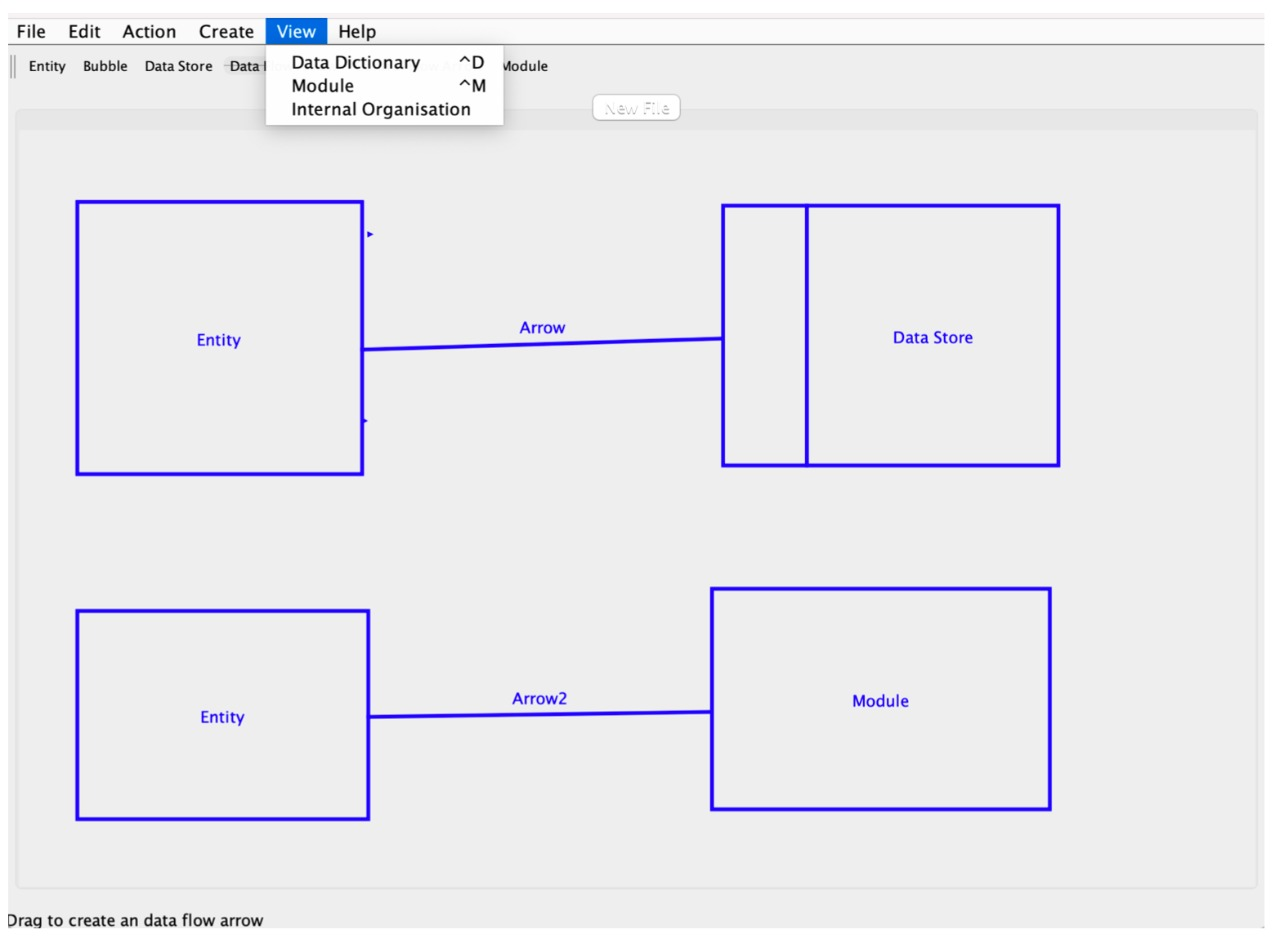


**ACTIVITY NETWORK**

****

****

## PROJECT DEMONSTRATION & RESULTS



The novelty of our project is that by using our CASE Tool we can draw efficient and quick DFDs very easily with a 97% accuracy rate. The user can access all the features of the tool with hints provided at the bottom of the tool. This tool can be applicable in various scenarios such as teaching, structured analysis, software design. We have also open source our project so that other interested people can contribute to the project to enhance its functionalities and build upon more features.

## CONCLUSION

Nowadays the development of software based systems is increasing exponentially due to its benefits. All the analysis and teachings are advancing and we need a better way to approach design. Hand Drawn DFDs and design ideas are alienating and in this digital era ,this CASE tool is efficient and easy to use with rich features that can sustain the user’s needs and grant an enriching experience to structured analysis and design .

## REFERENCES

1. Juan de Lara, Hans Vangheluwe (2002).[*Using AToM as a Meta-CASE Tool 2(8).*](https://repositorio.uam.es/bitstream/handle/10486/665853/atom_lara_ICEIS_2002.pdf?sequence=1)
2. Devdas Shetty ,Ahad Ali (2015). [*A new design tool for DFA/DFD based on rating factors*](https://www.emerald.com/insight/content/doi/10.1108/AA-11-2014-088/full/html)
3. D. Truscan, J.M. Fernandes, J. Lilius (2004). [*Tool support for DFD-UML model-based transformations*](https://ieeexplore.ieee.org/abstract/document/1316723)
4. Aamer Nadeem; Tai-Hoon Kim; Eun-Suk Cho, Atif Aftab Ahmed Jilani (2008). [*Formal Representations of the Data Flow Diagram: A Survey*](https://ieeexplore.ieee.org/abstract/document/4721332)
5. Max Breitling Jan Philipps. (2000). [*Diagrams for Dataflow.*](https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.29.7593&rep=rep1&type=pdf)