Design Justification

CS 345-346

Problem 2 : Classnote Taking Application

Group 18

Name	Roll Number
Param Aryan Singh	180101055
Parth Dhananjay Bakare	180101056
Pooja Gajendra Bhagat	180101057
Tejas Prashant Khairnar	180101081

Contents

1 Balance of Cohesion and Coupling Principles **1**

1.1 Cohesion of Modules 1

1.1.1 Module 1: Manage Users 1

1.1.2 Module 2: Manage Notebooks 1

1.1.3 Module 3: Edit Notes 1

1.1.4 Module 4: Examine Notes 2

1.2 Coupling between Modules 2

2 Conclusion **3**

1 Balance of Cohesion and Coupling Principles

The **Classnote Taking Application** is decomposed into four modules - Manage Users, Manage Notebooks, Edit Notes and Examine Notes.

1.1 Cohesion of Modules

1.1.1 Module 1: Manage Users

This module has three processes - Register New User, Login and Update Details.

- **Logical:** All the three processes cohesively perform the same functionality - managing user access to the application - while working on different aspects of it. When the application is first launched, either Register New User or Login is called to setup the account or provide access to an existing account, thereby performing similar tasks.
- **Temporal:** Register New User and Login are called at the same time - in the beginning.
- **Communication:** All the three processes deal with the same datastore - User Data. Register New User creates a new entry in User Data, Login refers existing entries in User Data to provide access to the account and Update Details edits an existing entry - it changes the username and/or the password in the entry.

1.1.2 Module 2: Manage Notebooks

This module has four processes - Create Notebook, Delete Notebook, Delete Note and Create Note.

- **Logical:** All the four processes deal with the structure and organization of notebooks, formulating it in conjunction to organize notes efficiently.
- **Communication:** Create Notebook and Delete Notebook modify Notebook Data, Create Note and Delete Note modify Notes Data.
- **Sequential:** Whenever we are deleting a notebook all the contents of the notebook i.e all the notes present in the notebook but also be deleted recursively.Hence there is output of Delete Notebook process acts as an input to the Delete Note process.

1.1.3 Module 3: Edit Notes

This module has five processes - Add Text, Add Image, Manage Table, Draw and Delete Content. Manage Table itself has four subroutines - Create Table, Edit Cell, Insert Row and Insert Column.

- **Logical:** Add Text, Add Image, Manage Table and Draw are tasked with inserting new content in the note, thus performing a similar operation. Within Manage Table, Insert Row and Insert Column help the table store more data.
- **Communication:** All the five processes deal with the same datastore - Notes Data. Add Text, Add Image, Manage Table and Draw create new entries or edit existing entries in Notes Data.

1.1.4 Module 4: Examine Notes

This module has four processes - Search by Text, Search by Date, Find in Note and View Note.

- **Logical:** Search by Text and Search by Date filter existing notes to return a list of selected notes. Both of them search for a certain keyword to perform the task.
  Find in Note too searches a keyword inside a note to display sections which contain the keyword.
- **Communication:** Search by Text and Search by Date refer Notebook Data to perform their tasks, whereas Find in Note and View Note access Notes Data for theirs.

1.2 Coupling between Modules

We have four main modules here:

- **Module 1:** Manage Users
- **Module 2:** Manage Notebooks
- **Module 3:** Edit Notes
- **Module 4:** Examine Notes

Coupling between modules pairwise:

- **Module 1 and Module 2:**
  - **Data:** **No**, The two modules do not communicate through any data item.
  - **Control:** **Yes**, The Notebooks/Notes cannot be created/deleted in Module 2 unless the user Logs-in into the application using process Login of Module 1.
  - **Content:** **No**, The two modules do not share any code.
- **Module 1 and Module 3:**
  - **Data:** **No**, The two modules do not communicate through any data item.
  - **Control:** **Yes**, The Notes cannot be edited in Module 3 unless the user Logs-in into the application using process Login of Module 1.
  - **Content:** **No**, The two modules do not share any code.
- **Module 1 and Module 4:**
  - **Data:** **No**, The two modules do not communicate through any data item.
  - **Control:** **Yes**, The Notes cannot be viewed or parsed for the user in Module 3 unless the user Logs-in into the application using process Login of Module 1.
  - **Content:** **No**, The two modules do not share any code.
- **Module 2 and Module 3:**
  - **Data:** **No**, The two modules do not communicate through any data item.
  - **Control:** **No**, The data from one module is not used to control the flow of instructions in other module.
  - **Content:** **No**, The two modules do not share any code.

- **Module 2 and Module 4:**
  - **Data: No**, The two modules do not communicate through any data item.
  - **Control: No**, The data from one module is not used to control the flow of instructions in other module.
  - **Content: No**, The two modules do not share any code.
- **Module 3 and Module 4:**
  - **Data: No**, The two modules do not communicate through any data item.
  - **Control: No**, The data from one module is not used to control the flow of instructions in other module.
  - **Content: No**, The two modules do not share any code.

2 Conclusion

We can observe from the above analysis that all four of our modules have a good amount of cohesion. Moreover, there are very few instances of coupling between the modules which cannot be reduced any further.
Hence, we have **high cohesion** and **low coupling** in our design, implying that our modules are functionally independent.