**The Story Line Tool**

**D20 2nd Semester Final Project**

A bird flying in the air

Description automatically generated with low confidence

EASV Sønderborg

Computer Science

Hand in: 17th June 2021

Authors: Andrej Simionenko, Raheela Tasneem, Fei Gu, Ibrah为eem Swaidan

Supervisors: Karsten Skov, Tommy Haugaard, Frank Østergaard Hansen

# Preface

preface

# Resource

* GitHub Link: <https://github.com/Fei-D20/D20_2nd_Semester_Final_Project.git>
* The project report: [https://github.com/Fei-D20/D20\_2nd\_Semester\_Final\_Project/blob/0a6c3eb512f8fc81127eb344f31c905944d0b893/Document/ProjectReport/Report.docx](https://github.com/Fei-D20/D20_2nd_Semester_Final_Project/blob/main/Document/ProjectReport/Report.docx)

Table of Contents

[PROJECT TITLE 1](#_Toc72313086)

[D20\_2nd\_Semester\_Final\_Project 1](#_Toc72313087)

[Preface 2](#_Toc72313088)

[Resource 2](#_Toc72313089)

[Introduction 6](#_Toc72313090)

[Case study 7](#_Toc72313091)

[Requirement study 8](#_Toc72313092)

[Delimitations 9](#_Toc72313093)

[Technology and Business 10](#_Toc72313094)

[The Business Model Canvas 10](#_Toc72313095)

[Competitor Examples 11](#_Toc72313096)

[Values for the user 12](#_Toc72313097)

[System development 13](#_Toc72313098)

[Unify process 13](#_Toc72313099)

[Inception phase 13](#_Toc72313100)

[Elaboration phase 13](#_Toc72313101)

[Construction 13](#_Toc72313102)

[Transition 13](#_Toc72313103)

[Domain analyses 14](#_Toc72313104)

[Vision 14](#_Toc72313105)

[Data dictionary 14](#_Toc72313106)

[Domain model 14](#_Toc72313107)

[UP 15](#_Toc72313108)

[Process flow 15](#_Toc72313109)

[Use Cases 15](#_Toc72313110)

[Time estimation 17](#_Toc72313111)

[Throw away prototyping 17](#_Toc72313112)

[Usability 17](#_Toc72313113)

[Class diagram 17](#_Toc72313114)

[SSDs 17](#_Toc72313115)

[Inspection 17](#_Toc72313116)

[Reflections 17](#_Toc72313117)

[Recommendation of improvements 17](#_Toc72313118)

[Programming 18](#_Toc72313119)

[Architecture(diagram) 18](#_Toc72313120)

[Patten analyses 19](#_Toc72313121)

[Database analyses 20](#_Toc72313122)

[Database Model (ERD) 20](#_Toc72313123)

[Database relationship 20](#_Toc72313124)

[Database creation (scripts ) 20](#_Toc72313125)

[JDBC 20](#_Toc72313126)

[Stored procedures 20](#_Toc72313127)

[GUI design 21](#_Toc72313128)

[GUI structure 21](#_Toc72313129)

[GUI controller 21](#_Toc72313130)

[Unit-test 21](#_Toc72313131)

[Design Class Diagram 22](#_Toc72313132)

[Sequence diagram 23](#_Toc72313133)

[Conclusion 24](#_Toc72313134)

[Bibliography 25](#_Toc72313135)

[Appendices 26](#_Toc72313136)

Error! Hyperlink reference not valid.Error! Hyperlink reference not valid.Error! Hyperlink reference not valid.Error! Hyperlink reference not valid.Error! Hyperlink reference not valid.Error! Hyperlink reference not valid.Error! Hyperlink reference not valid.Error! Hyperlink reference not valid.Error! Hyperlink reference not valid.Error! Hyperlink reference not valid.Error! Hyperlink reference not valid.Error! Hyperlink reference not valid.Error! Hyperlink reference not valid.Error! Hyperlink reference not valid.Error! Hyperlink reference not valid.Error! Hyperlink reference not valid.Error! Hyperlink reference not valid.Error! Hyperlink reference not valid.Error! Hyperlink reference not valid.Error! Hyperlink reference not valid.Error! Hyperlink reference not valid.Error! Hyperlink reference not valid.Error! Hyperlink reference not valid.Error! Hyperlink reference not valid.Error! Hyperlink reference not valid.Error! Hyperlink reference not valid.Error! Hyperlink reference not valid.Error! Hyperlink reference not valid.Error! Hyperlink reference not valid.Error! Hyperlink reference not valid.Error! Hyperlink reference not valid.Error! Hyperlink reference not valid.

## The tabels reference

[Table 1: Development case 13](#_Toc72313383)

[Table 2: The use case 15](#_Toc72313384)

## The Figures reference

[Figure 1 : Domain model 12](#_Toc72661897)

[Figure 2: Architecture Diagram 20](#_Toc72661898)

# Introduction

//

# Case study

Our team has been tasked with creating a tool, that would help writers composing a new book or a story. Currently, many authors begin their new work by collecting many notes about the story and its proceedings, often written on post-it notes or event cards. This is where the problems arise: when there are a lot of cards its quite hard to manage and keep track of all of them, they can be easily misplaced or lost, moreover it is hard for multiple authors to work with them, it takes a long time to sort them and find a needed one to create a flowing storyline. This is where the Story Line Tool comes in, it will prevent all the issues mentioned above and simplify the work of an author. The tool will allow a single or multiple authors to easily work on a project by allowing them to take and save notes, rearrange them in desired order, print the events in chronological order and export everything in a text file.

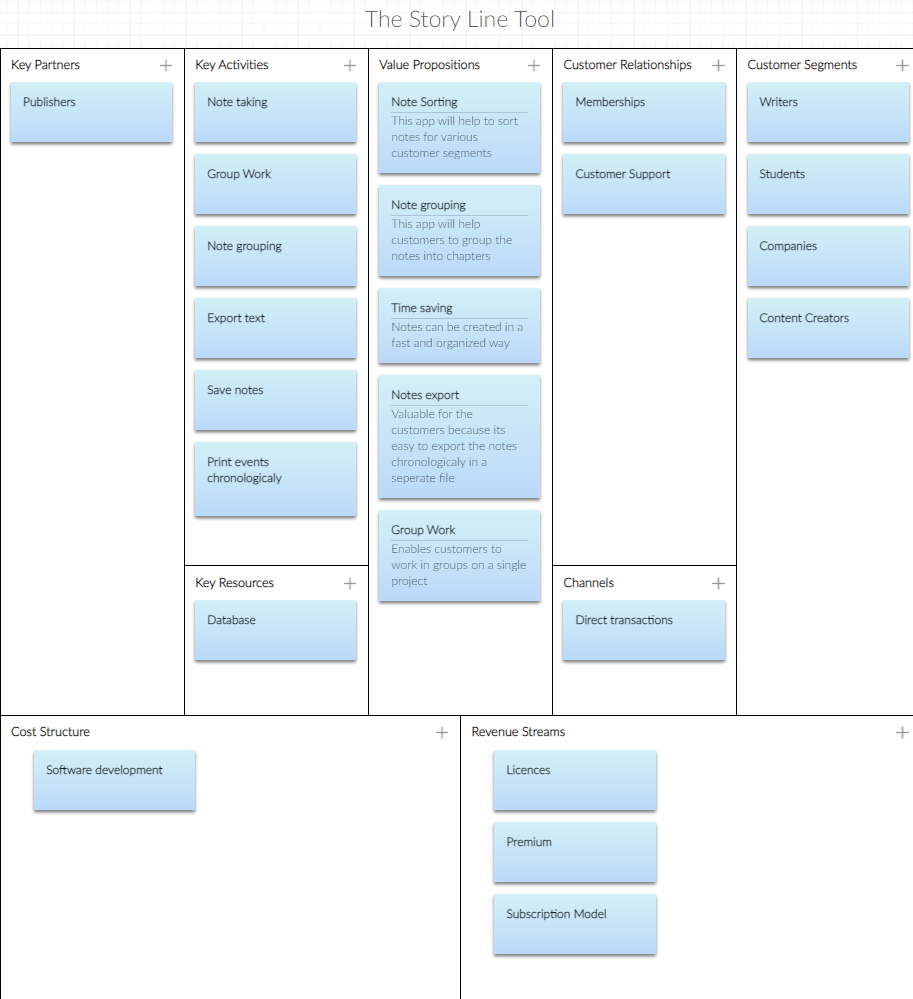
Our team believes that such tool would be very useful for authors, to ease their process of creation, and that it has a place in the market.

# Delimitations

Delimitations

# Technology and Business

## The Business Model Canvas



## Competitor Examples

## Values for the user

# System development

## Domain analyses

### Vision

This application is working for the author who need to have a platform to store, organize, manage the ideas, and also cooperation with some other authors. They can share the idea between multiple cooperators and leave some opinions as well. And also, they can use this application create chronicle of events, and check the view of the story timeline. Make the decision about the different choice which is the actor made. And check how is going on.

This application can also export the events as the text file with formatter.

### Data dictionary

***Application:*** the software we are trying to make behand this report.

***The author:*** the final user for this application who want to get help for to make event card and easier cooperation work.

***Event card:*** the small note (like sticky note or tile) on the working board with single words. And the user can create, delete, edit, search and sort it.

***Event:*** the situation or action the role play in the story.

***Sub-event:*** the individually event which under the event as an element.

***Relationship:*** the connected between different events.

***Role:*** the character who are in the story.

***Event time:*** the time which event happen in the story.

***Note:*** the event detail.

***Chapters:*** the categories or sections of story which is the event lay on.

***Comments:*** the author leave some description about event (or make a note for the event detail) and also for multiple authors leave notes to each other.

***Story:*** the novel author will write when after creating all event and timeline.

***Timeline:*** the story event flow following the time.

***Direction:*** what will happen in the story.

***Levels:*** some event card under another event.

***Events chronologically:*** the event happen following the time.

### Domain model

The domain model descripts about what kind of entity in this domain area, and which attribute they have. Also include about the relationship between the entity.

In this case, an author or multiple authors who have own name can write some events.

A lot of events can constitute one story. And a different direction of story creates different timeline in the story.

Otherwise, the author can leave some note and comment under the event card

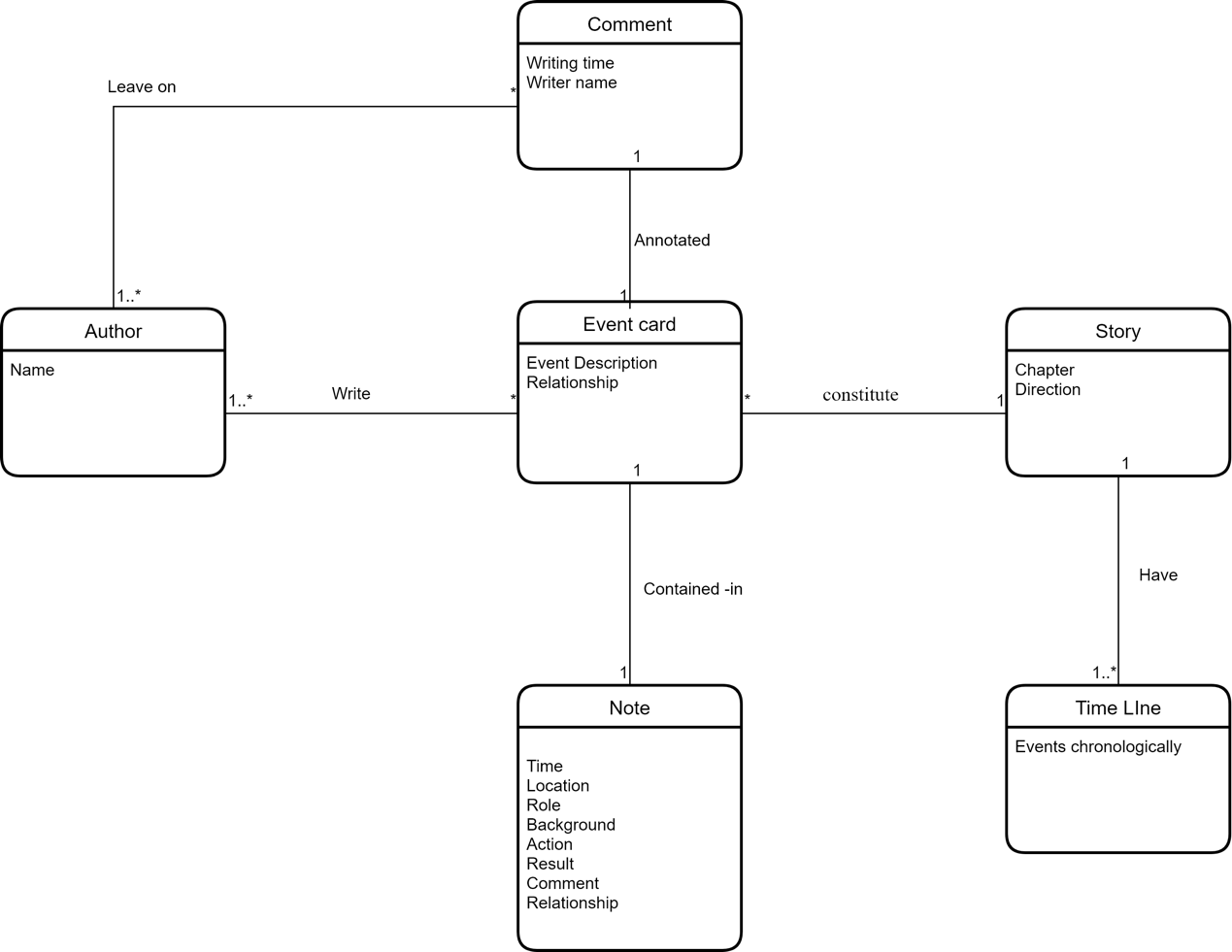


Figure 1 : Domain model

## Artifacts in UP

Table 1: Development case)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Discipline | Artifact | Comment | Inception phase | Elaboration phase | Construction | Transition |
| Technology and business | Canvas |  | Start | Revise |  |  |
|  | Competitor Examples |  | Start |  |  | Revise |
|  | Values for the user |  | Start |  |  | Revise |
| System development | Vision |  | Start | Revise |  | Revise |
|  | Data dictionary |  | Start | Revise |  |  |
|  | Domain model |  | Start | Revise |  |  |
|  | Use case |  | Start | Revise |  | Revise |
| Program Design | Architecture |  | Start | Revise |  |  |
|  | Class diagram |  |  | Start | Revise |  |
|  | SSDs |  |  | Start | Revise |  |
|  | SD |  |  | Start | Revise |  |
| Program implementation | Source Code |  |  |  | Start | Revise |
|  | Java Docs |  |  |  | Start | Revise |
|  | Database |  |  |  | Start | Revise |
|  | GUI |  |  |  | Start | Revise |
| Testing | Test |  |  |  | Start | Revise |

The artifacts in UP is discussing about the different part in this project which are located on the particular phase and identify when we should check and improve it more.

From the beginning we have to see from the angle of the Technology and Business discipline. That will be inception phase.

## UP

### Use case

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Use case number | Actor | Use case | Version | Time Estimation |
| UC 1 | User | Register as new user | 1 | 12 Hours |
| UC 2 | User | Create User login | 1 | 8Hours |
| UC 3 | User | Create password field | 1 | 6Hours |
| UC 4 | User | Forgot Password | 1 | 6Hours |
| UC 5 | User | Create event box | 1 | 12Hours |
| UC 6 | User | Create store event option / button | 1 | 6hours |
| UC 7 | User | Make connection between events | 1 | 6Hours |
| UC 8 | User | Set the priority about events (or level as well) | 1 | 6Hours |
| UC 9 | User | Set a comment about event card | 1 | 8Hours |
| UC 10 | User | Sort the chosen event/all event | 1 | 8Hours |
| UC 11 | User | Make Store event option/ button | 1 | 8Hours |
| UC 12 | User | Create Search event card | 1 | 12Hours |
| UC 13 | User | Insert some event (from .TXT or .md file) into a project | 1 | 10Hours |
| UC 14 | User | Create a new note | 1 | 12 Hours |
| UC 15 | User | Store/ save note option | 1 | 6 Hours |
| UC 16 | User | Create Delete note option | 1 | 6 Hours |
| UC 17 | User | Create Drag and drop sticky notes option | 1 | 12Hours |
| UC 18 | User | Edit a note option | 1 | 6Hours |
| UC 19 | User | Add a sub-note | 1 | 8 Hours |
| UC 20 | User | Create Categories option (like chapters) | 1 | 8 Hours |
| UC 21 | User | Create a redo option | 1 | 6 Hours |
| UC 22 | User | Create undo option | 1 | 6 Hours |
| UC 23 | User | Create change color option | 1 | 8 Hours |
| UC24 | User | Create a new story line | 1 | 12 Hours |
| UC25 | User | Modify the story line | 1 | 8 Hours |
| UC 26 | User | Mention Time | 1 | hours |
| UC 27 | User | Create a Timeline | 1 | 15 Hours |
| UC 28 | User | Create a project (novel) | 1 | 8 Hours |
| UC 29 | User | Share a project to another author  (Or choose/add another user as cooperator) | 1 | 6 Hours |
| UC 30 | User | Save project | 1 | 4 Hours |
| UC 31 | User | Export a project as text (like .TXT file or .md file) | 1 | 6 Hours |
| UC 32 | User | Import a project as text | 1 | 8 Hours |
| UC 33 | User | Create View Option (list or gallery) | 1 | 10Hours |
| UC 34 | User | Create date-based reminder | 1 | 12 Hours |
| UC 35 | User | Change project name | 1 | 6Hours |
| UC 36 | User | Create choose font and size option | 1 | 8 Hours |
| UC 37 | User | Add Picture | 1 | 6 Hours |
|  |  |  |  |  |
| Database |  |  | 1 | 30hours |
|  |  |  |  |  |
| Software to connect database and use cases |  |  |  | 30 hours |
|  |  |  |  |  |

Table 2: The use case

Casual use cases:

|  |  |  |  |
| --- | --- | --- | --- |
| Use case ID | Use case name | Actor | Flow of events |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Fully dressed use cases:

|  |  |
| --- | --- |
| Use Case Id | UC1 |
| Trigger | The user wants to create a new note |
| Actor | User |
| Version | 1 |
| Flow of events | 1. The use case starts when the user clicks on “create” button 2. The actor writes a note 3. The actor clicks on “save” button to save 4. The actor drags and drop the note in the right order(event) |
| Date | 21/05/21 |
| Author | IS |

|  |  |
| --- | --- |
| Use Case Id | UC18 |
| Trigger | Register as new user |
| Actor | User |
| Version | 1 |
| Flow of events | 1. User select “sign up”  2. User fill the necessary information  3. User clicks on “Accept” to execute the registration |
| Date | 21/05/21 |
| Author | IS |

### Time estimation

### Throw away prototyping

### Usability

### SSDs

### Inspection

During the implement with architecture, the data dictionary has some disorder part and unclear the meaning. So we remake and organize some.

### Reflections

### Recommendation of improvements

# Programming

Programming

## Architecture

The architecture is the basic constructure about the application. It defined the modules and the packages and identify the relationship and connection between the modules and the packages.

### Domain module

Base on this case, the most first module should be **Domain module**. The domain module is following the Domain model. It transfers from the real word entities to the oriented object. Domain modules include 3 packages in this case:

1. **Author**: The end user for this application which include the authors declare.
2. **Event** **card**: The piece of event which the author writes. Also include the *Event*, the *event time*, the *role*, the *note*, the *sub-event*, the *chapter*, the *comments*.
3. **Story**: The collection of the event, and are organized by event time, direction, and so on. Include in *timeline, direction, levels, relationship.* At the end, showing on *events chronologic.*

### Function module

The **Function module** is the function which the Domain objects possess for implement the method. Which means the function module will based on the domain module and control the object to do something and reached some result.

This module will include same 3 package as domain module:

1. **Author Function**: Which has the author register, author log in/out, remove author, and so on.
2. **Event Card Function**: This package is controlling the event card and modify it. It will process create event card, delete event card, back up event card, edit event card, etc.
3. **Story Function**: Story function will include the Story organize function, and out put the events chronologically.

### GUI module

GUI is the user interface for the end user operate the whole application. It will have 3 packages following the domain module same:

1. **Author UI**
2. **Event Card UI**
3. **Story UI**

### Application module

Application module is the connection between GUI and function module. In another word, the application can also call as controller module. So when the user for example click a button the method belong to GUI will invoke the method belong to application module and to reach the function method in the function module. So obesely, the application module also should have same 3 package each to each:

1. **Author controller**
2. **Event Card controller**
3. **Story controller**

### Database module

The database module is the only on not following the Domain module because the database just needs 2 package which is **database connection util** for create the connection between application and database and the **CRUD** package to modify the database during we have cooperation author.

The diagram about architecture following is showing up the basic relationship between the module and packages.

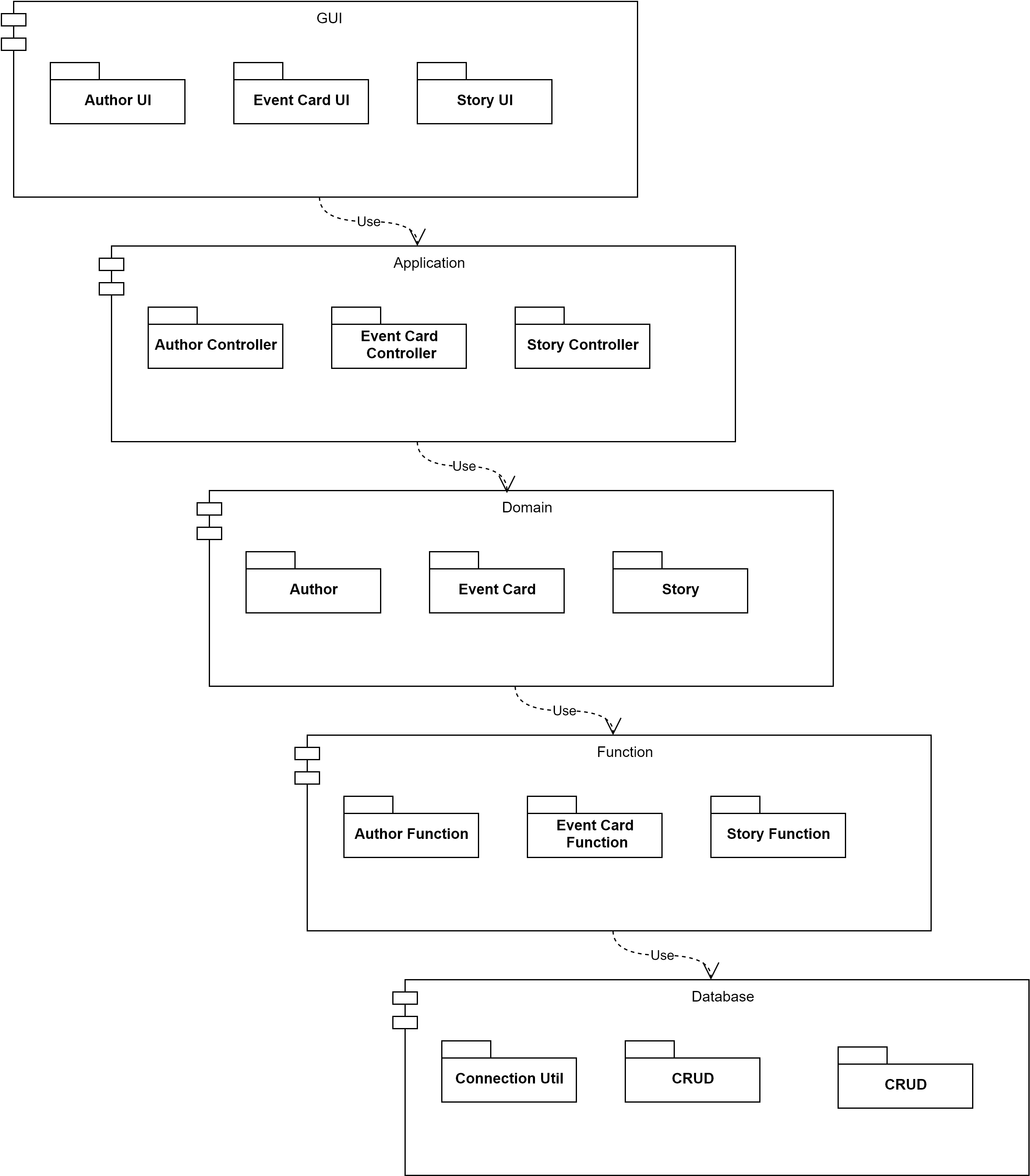


Figure 2: Architecture Diagram

## Class Diagram

The class diagram is based on the architecture and extend it on the class level. Therefore, the class diagram can follow the structure of architecture. And make few different diagrams.

### Domain:

In the domain package, the class include about the object are transferred from domain model and include the property of the object.

#### Domain: Author

The author package process **user interface** and **the author class**. And the user interface is used for reserve further such as editor or publisher. The author class declare the object author, and the author property, name etc. And it will implement the user interface.

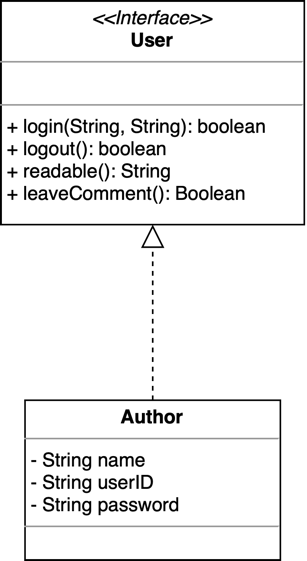


Figure: Author class diagram

#### Domain: Event Card

The event card is the most important package in this application. It includes all the core part such as the event and the relationships.

#### Domain: Story

### Function:

#### Function: Author Function

#### Function: Event Card Function

#### Function: Story Function

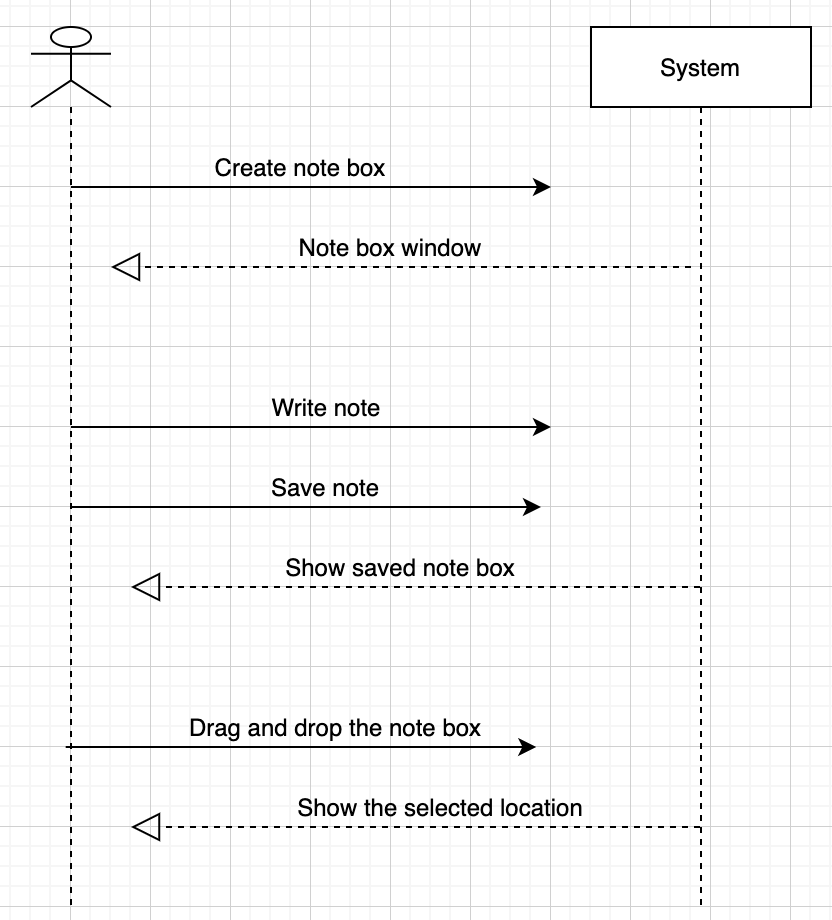
### Application:

### GUI:

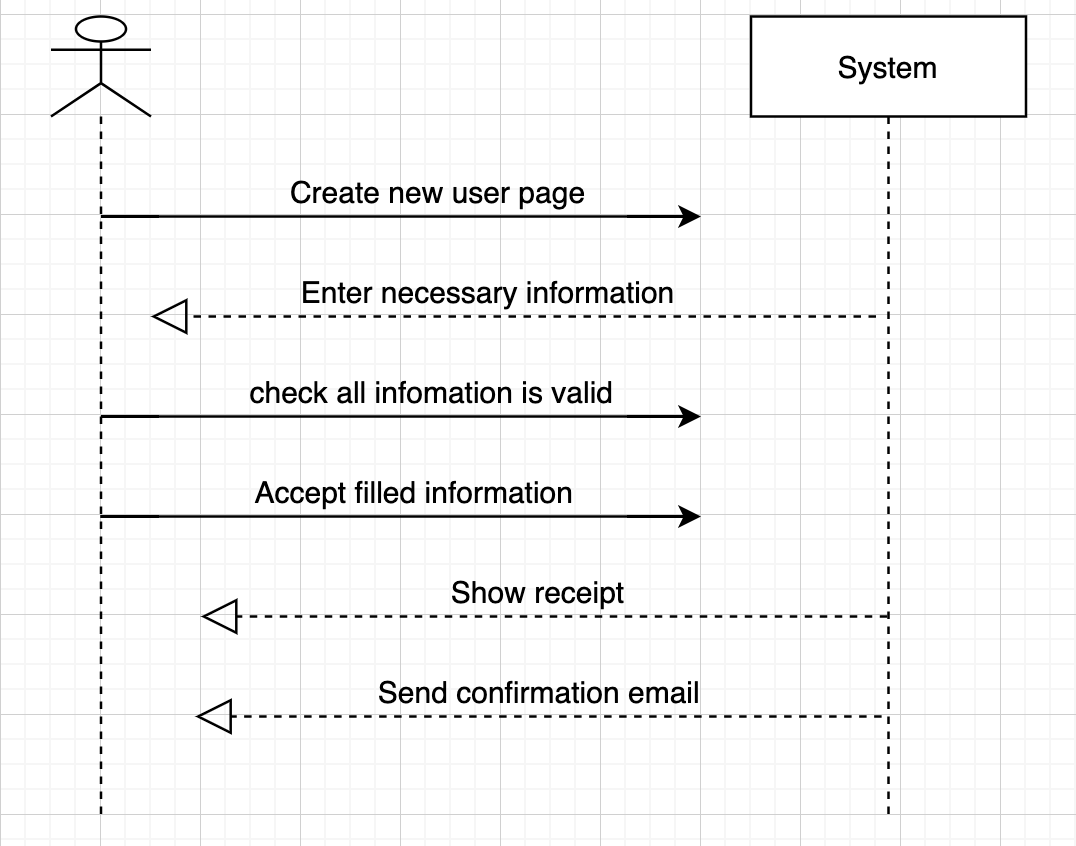
### Database:

Sequence diagram Sequence diagrams is created to understand how the program handles specific events. The System Sequence diagram describes an event that the system needs to process. This System Sequence diagram provides a visual representation of how the system will handle the UC18 create user process and the UC1 create note box process. An actor sends information to the system, then the system processes the data and sends the information back to the actor.

UC1



UC18



## Pattern analyses

## Database analyses

### Database Model (ERD)

### Database relationship

### Database creation (scripts )

### JDBC

### Stored procedures

## GUI design

### GUI structure

### GUI controller

### Unit-test

# Conclusion

Conclusion

# Bibliography

Bibliography

# Appendices

Appendices