

FlashPig SDD

Jesper Bergquist

Wendy Pau
Salvija Zelvyte

Madeleine Xia

October 16, 2020

Contents

| | | |
|----------|--|----------|
| 1 | Introduction | 1 |
| 1.1 | Definitions, acronyms, and abbreviations | 1 |
| 2 | System architecture | 1 |
| 2.1 | Dependency analysis | 2 |
| 3 | System Design | 2 |
| 3.1 | MVVM | 2 |
| 3.2 | Design Patterns | 3 |
| 3.2.1 | Adapter pattern | 3 |
| 3.2.2 | Observer pattern | 3 |
| 3.2.3 | Singleton pattern | 3 |
| 3.3 | UML sequence diagram | 3 |
| 4 | Persistent data management | 3 |
| 5 | Quality | 3 |
| 5.1 | Access control and security | 3 |
| 6 | References | 3 |

1 Introduction

Flashpig is an application that provides the opportunity to create flashcards and use them to study and learn in three different ways.

The first way is "Flashcard" which consists of two sided cards, where one side has a question while the other has an answer, both sides can be appeared as picture or text. After each card the user has to decide which degree of difficulty the card had (easy, medium or hard). This makes the learning progress visible and accessible for the user. Furthermore, depending on what difficulty the user chose the cards will be shown in different interval of time which is based on an algorithm. As a result the "flashcard progress" will be based on the chosen complexity and will be available on the homepage for the user to check their score.

In addition there is also two other games named "Memory" and "Pair up". "Memory" works exactly like the classic memory game where in this case you have to combine front- and backside. However, "Pair up" makes all the cards revealed and gives the user multiple possible answers to match with the questions. Therefore the front sides (questions) is located on the left column and backsides (answers) on the right, all cards are in random order to increase the difficulty for the user.

The application is mainly designed for students but will work for any other user that want to learn new information in a quick, effective and fun way.

1.1 Definitions, acronyms, and abbreviations

2 System architecture

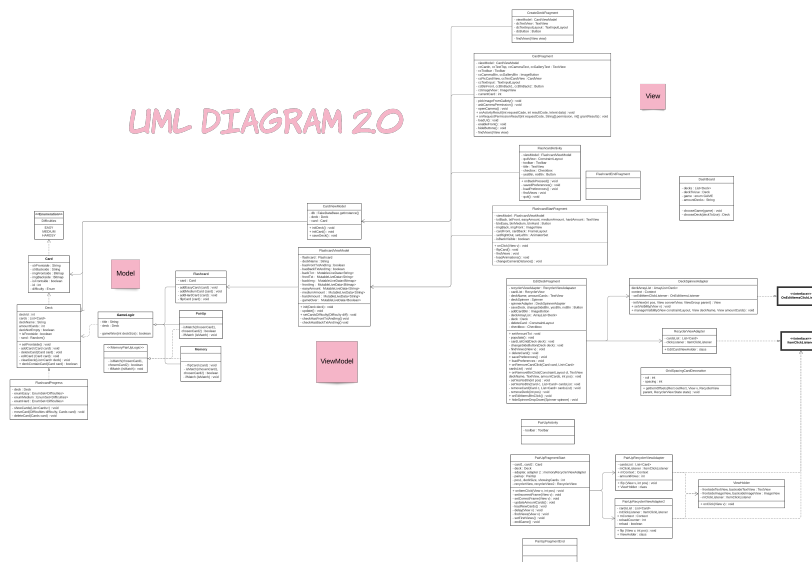
The most overall, top level description of your application. If your application uses multiple components (such as servers, databases, etc.), describe their responsibilities here and show how they are dependent on each other and how they communicate (which protocols etc.) You will to describe the 'flow' of the application at a high level. What happens if the application is started (and later stopped) and what the normal flow of operation is. Relate this to the different components (if any) in your application.

2.1 Dependency analysis

3 System Design

Draw an UML package diagram for the top level for all components that you have identified above (which can be just one if you develop a standalone application). Describe the interfaces and dependencies between the packages. Describe how you have implemented the MVC design pattern. Create an UML class diagram for every package. One of the packages will contain the model of your application. This will be the design model of your application, describe in detail the relation between your domain model and your design model. There should be a clear and logical relation between the two. Make sure that these models stay in 'sync' during the development of your application. Describe which (if any) design patterns you have used. The above describes the static design of your application. It may sometimes be necessary to describe the dynamic design of your application as well. You can use an UML sequence diagram to show the different parts of your application communicate an in what order.

3.1 MVVM



Model

ViewModel

View

3.2 Design Patterns

3.2.1 Adapter pattern

3.2.2 Observer pattern

3.2.3 Singleton pattern

3.3 UML sequence diagram

4 Persistent data management

If your application makes use of persistent data (for example stores user profiles etc.), then explain how you store data (and other resources such as icons, images, audio, etc.).

5 Quality

Describe how you test your application and where to find these tests. If applicable, give a link to your continuous integration. List all known issues. Run analytical tools on your software and show the results. Use for example: – Dependencies: STAN or similar. – Quality tool reports, like PMD. NOTE: Each Java, XML, etc. file should have a header comment: Author, responsibility, used by ..., uses ..., etc.

5.1 Access control and security

If your application has some kind of access control, for example a login, or has different user roles (admin, standard, etc.), then explain how your application manages this.

NA

6 References

List all references to external tools, platforms, libraries, papers, etc. The purpose is that the reader can find additional information quickly and use this to understand how your application works.