CS 319 - Object-Oriented Software Engineering

Final Report

CSCrush

Group 1A

Taner Düzceer

Eren Ayture

Ahmet Akif Uğurtan

Berk Ataç

1. Introduction

There are a lot of puzzle video games out there with different kind of objectives. CSCrush is a 2-D match-three puzzle type video game we decided to develop as desktop application. The user can play this game by using mouse. In our game, user try to reach a specific points which is given by the game according to the level. The user should change the places of the icon of different type of books. The user can change the place of books if the books are neighbor each other vertically or horizontally. The user tries to catch at least three the same order of same type of books vertically or horizontally. When the user can order like this, the user get points and the ordered books are disappeared. Therefore, this game tests the user's problem solving skills. When the user pass a level of this game, another level is opened. This level is more difficult than the previous one. Moreover, there are some power-ups which are help the user. The user can use them in difficult circumstances.

The game we were influenced by is Candy Crush.https://king.com/tr/game/candycrush CSCrush will not be an exact replica of Candy Crush. In CSCrush classic matching-three game rules will apply. Player will be presented with a level full of objects and try to match as many as possible, using various power-ups, in order to earn score then advance to the next levels and there will be finite amount of moves for the player. Players who complete levels will be on high score board.

2. Overview

CSCrush is themed after CS department in Bilkent University. All objects which player will be matching are CS course books and all power-ups which are there to help the player, make gameplay fun and fast paced will be Bilkent University CS course instructers. Player will complete levels by swapping course books to make a match of three or more of the same books, eliminating those books from the game board and replacing them with new ones, which could possibly create more matches. Matches of four or more books create unique books that act as power-ups with larger board-clearing abilities. Goal of CSCrush is making as many matches as possible within finite amount of moves.

2.1 Gameplay

Player will need a mouse to play the game. Player will enter their user name and password using his/her keyboard to login to the game. Player will swap course books using mouse and match three or more of the same book in order to eliminate them. New books will drop from above to replace eliminated books. Matches of four or more will create special books which can eliminate a row or a column of books. For every level, player will have a finite amount of swappes to perform. Player will have access to certain power-ups and with the help of these power-ups player will earn score points for every elimination. Power-ups can be used by clicking on them with left mouse button.

Player will have to achieve a certain amount of score on each level within their finite amount of swapps in order to advance to the next level.When player completes a level, achieved score on that level will be added to the total score of that player and that total score will appear on high score board with user name player uses to login.

2.2 Levels

There will be ten playable levels in CSCrush. On each level there will be 10x10 gameboard in which all course books will be. On some levels not all spots on gameboard will be movable.

2.3 Power Ups And Their Usage

In this game, we will have power ups consist of teachers, Since project is cs-crash, teachers are going to in our department. Teachers are used as buttons with their images. Those buttons are going to be responsible for different features in this game. This power ups will ease to help devastating courses. Therefore, user will gain more points. However, using these power ups diminishes the points up to the teachers and their features.

2.3.1 Halil Altay Guvenir

As Halil Altay Guvenir is our department chair, he is the most powerful power up. When this button is a hit, 5 courses(Topic 2.4) transform in to special courses that can clear an entire row or column of courses.

(figure 1)

2.3.2 William Sawyer

This power up requires users to select a course from inside suitable 3x3 center. If this button is a hit, and user selects the course 3x3 matrix of courses are removed.

(figure 1.1)

2.3.3 Robin Ann Downey

This power up requires users to select a course from any place in the matrix. The chosen course randomly alternate to the one of the courses. However, it could be the same as selected course.

(figure 1.2)

2.3.4 Ozcan Ozturk

This is a key power up. It can clean a selected course. One column may fall and leads to destruction of other courses above. Then, immense point might be gained by user.

(figure 1.3)

2.3.5 Eray Tuzun

This is one of the most important power up. User chooses two coursen in the matrix. These courses replaced with one another. It might lead to destruction of two different zones in the matrix, so result could be devastating.

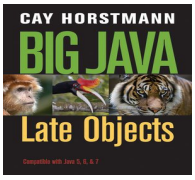
(figure 1.4)

2.4 Courses

In our 10x10 matrix in each matrix is filled with the course objects. These objects are consist of course book images. There are five of them to represent different courses. However, it does not mean that they have different features. If from horizontal or vertical aspects 3 or more indistinguishable objects come together, those course objects will be removed and user will gain point. Removed places filled with existing courses above, if there is no object above, courses created randomly to these certain places. Also, after removal of courses there is still empty spaces in the matrix, still courses are randomly created. There will be 5 objects with there type.

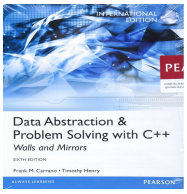
2.4.1 Course Types

2.4.1.1 CS-102 Algorithms and Programming II



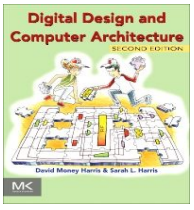
(figure 2)

2.4.1.2 CS-201 Fundamental Structures of Computer Science I



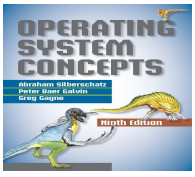
(figure 2.1)

2.4.1.3 CS-224 Computer Organization



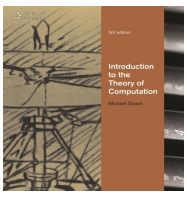
(figure 2.2)

2.4.1.4 CS-342 Operating Systems



(figure 2.3)

2.4.1.5 CS-476 Automata Theory and Formal Languages



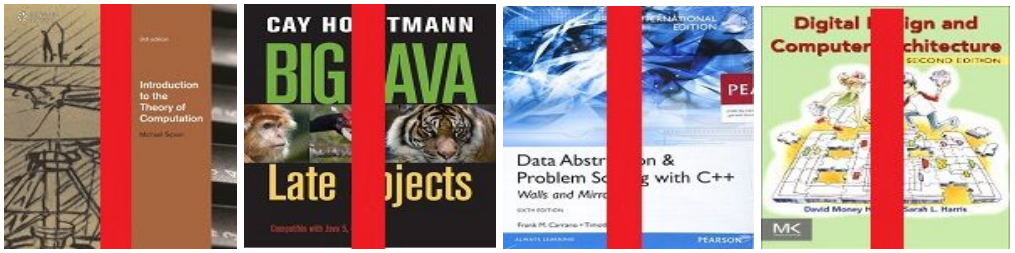
(figure 2.4)

2.4.2 Five Courses With Lines

These type of course objects cleans its row or column of other objects in the matrix. There is no need for gathering same group of course objects while removing process. Then empty places are filled as mentioned(Topic 2.4).

2.4.2.1 Five Courses With Vertical Lines

When These Objects are hit, it removes all the vertical line of its posizion.



(figure 3)

2.4.2.2 Five Courses With Horizontal Lines

When These Objects are hit, it removes all the horizontal line of its posizion.

(figure 4)

3. Requirement Specification

It will define the game in terms of user perspective by dividing functional, non-functional and pseudo groups.

3.1 Functional Requirements

It will describe the interactions between the game and the user.

3.1.1 Log in

The user will log in after entering the username and password. The system will check the credentials to provide security.

3.1.2Sign Up

lf the user wants to play this game first time, he/she must determine the username and password. The system will register the user.

3.1.3 Play Game

The game will start after play game button is pushed. Different levels in order to be chosen will appear as a grid menu. Player will choose one of levels to play first time or to pass his/her highscore. After choosing level, 10x10 matrix map will appear with random or fixed located objects such as cs courses, unmovable objects, teleport objects etc.. CS courses will get crushed and disappeared by swapping two adjacent objects and making a line composed of adjacent at least three same courses. If four adjacent same courses crushes at the same time, player will get a powerful course which destroys a whole row or column of courses if it gets crushed at the location of swapping. If five adjacent same courses crushes at the same time player will get a perfect object which destroys every similar objects swapped with it at the location of swapping. New course objects will fall from upside of matrix as courses get crushed. Boosters as named Bilkent CS instructors will appear at the bottom of screen and they are being able to use during game to crush objects easily in different ways. After player have reached the target point, a pop-up will appear by indicating whether player got a new high score or not and also asking for restarting.

3.1.4 Settings

There will be two properties to set. Music sound will be turned off via switch. Also users will log out via log out button.

3.1.5 Help and Information

There will be an explanation with images as a tutorial in order to show how to crush courses, what features boosters have, what every object means. This page will be accessed through main menu or during playgame screen. It will also give information about CS courses and instructors in real life since the objects and boosters are imitation of them.

3.1.6 High Score

The button of this window is in main menu window. The gamer should push the button to access high score window. In this window, the gamer can see 10 highest scores in order from small to large and a back button to return to the previous menu. The gamer should enter user name and password to see the scores. The gamer's own score can also be seen the window to compare it with the the highest scores. Finishing all levels in the game is not necessary to place in high score window because the gamer can collect enough point to place there. The high score list will be shown by using database. So, this list will updated and stored.

3.1.7 Credit

The access button of this window will be in menu window. The gamer can see the list of developers and a back button to return to the previous menu. The developers' contact information will be there. The gamers can send comments and suggestions to developers by using this window.

3.1.8 Exit

When the gamer push this button, the game terminates itself.

3.2 Non Functional Requirements

• The game graphics will be attractive. The objects will be chosen correctly to attract the gamer's attention. The chosen books will change the places in attractive animation.

• There will be no delay more than two seconds.

• There will be a notification which pops up to inform the user about exceptions and errors.

• The game can work on any machine which can run Java 8.

• None of the private information will be shared with public.

• To preserve the ear health, the music sound decibel cannot be greater than 90 dB.

• User has to have internet connection to play game because the game will save data in database server and will get data from database.

• There is no age limitation to play the game.

3.3 Pseudo Functional Requirements

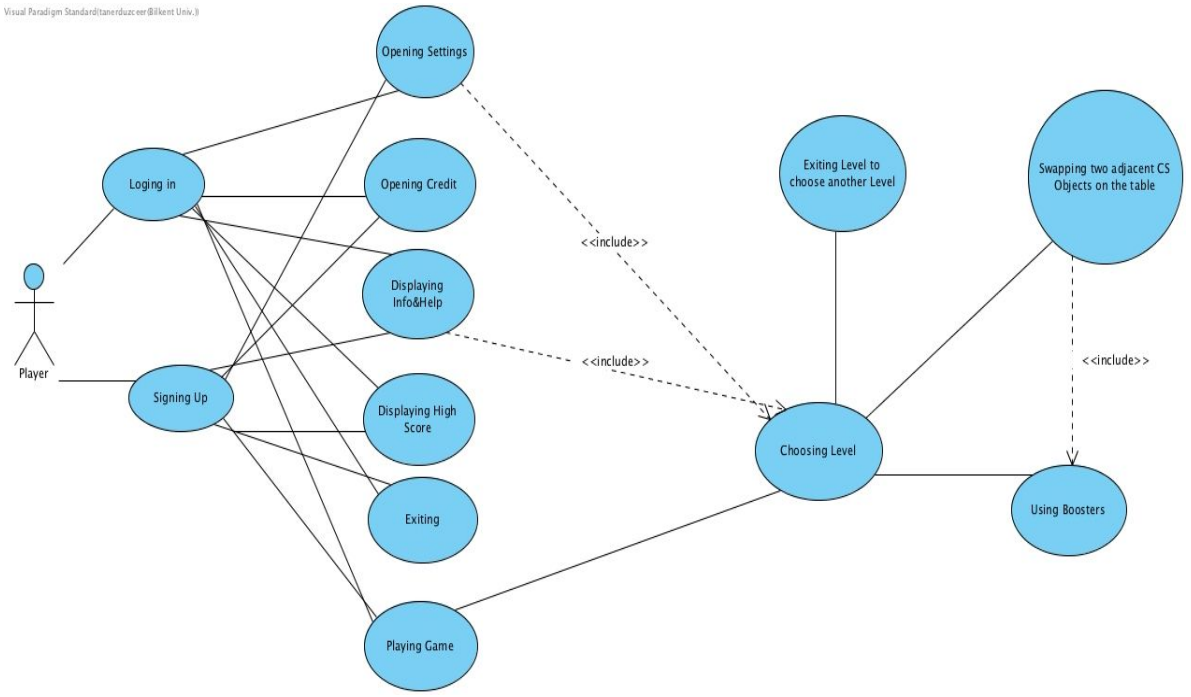
The game will be implemented in Java 8.

The user interface parts will be done by using Balzamiq 3.5.15, Visual Paradigm 14.8.

The database part will be managed by using MYSQL 5.7.

4. System Model

This section provides information about the main use case model of CSCrush game, detailed use case explanations are included below.



(figure 5)

4.1 Use Case Model

4.1.1 Play Game

Use case 1: Play Game

Actors involved: Player

Aim and System Response:

- Player aims to choose and swap objects efficiently and earn score in order to go next level before they run our of moves. Running out of means game is finished.

-System keeps the score of the Player.

Entry condition: Player has opened the game and selected Play Game from the main menu.

Exit condition:

1. Player has finished level reaching a certain score before they run out of moves, OR

2.. Player has run out of moves and did not have enough score to pass next level, OR

3. Player has exited the game from the in-game menu.

Preconditions: Player defined settings are used when starting a game if the player hasn't visited & changed the settings tab from the main menu, pre-defined settings are used when Play Game is selected.

Post-conditions: If the player's score is high enough to be on the leaderboard, it is added to the highest scores list.

Event Flow:

1. The player starts the game from the first level.

2. Player begins to swap objects.

Flow A:

3A. Player successfully finishes all the levels.

4A. The player's score is displayed on screen. If the player is among the top ten scorers, he enters his name to be added to the high scores list.

5A . Player returns to the main menu.

Flow B:

3B. Player can not collect enough points to advance next level.

4B. The player's score is displayed on screen. If the player is among the top ten scorers, he enters his name to be added to the high scores list.

5B. Player returns to the main menu.

Flow C:

3C. Player pauses the game while playing, accessing the in-game menu.

4C. Player decides to quit from the in-game menu before finishing the game.

Flow D:

3D. Player decides to continue playing.(Back to step 2)

4.1.2 Login

Use case 2: Login

Actors Involved: Player

Aim and Response: Player wants to have access to game main menu in order to change settings, view help or credits, play game, check high scrores or submit score.

Pre-condition: Player does not have access to main menu can not submit score.

Post-condition: Player has access to main menu and can submit high score.

Entry Condition: Player selects "Login" button from screen.

Exit Condition : Player logs in or selects "Back" button.

Event Flow :

1. Player clicks "Login" button.
2. UserID and Password spaces are displayed.

Flow A :

3A: User already has an account so he/she enter required info.

Flow B:

3B: Player does not have an account so he/she exits.

4B: Player heads on to the "Sign Up" page.

Flow C:

3C: Player exits with out logging in.

4.1.3 Sign Up

Use case 3: Sign Up

Actors Involved: Player

Aim and Response:

Player wants to create an account so he/she have access to game main menu in order to change settings, view help or credits, play game, check high scrores or submit high score.

Pre-condition: Player does not have an account..

Post-condition: Player has an account.

Entry Condition: Player selects “Sign Up” button from screen.

Exit Condition : Player creates an account or selects “Back” button.

Event Flow :

3. Player clicks “Sign Up” button.
4. Account creation page opens.

Flow A :

3A: User already has an account so he/she exits..

Flow B:

3B: Player does not have an account so he/she fills required information..

Flow C:

3C: Player exits with out signin up.

4.1.4 Change Settings

Use case 4: Change Settings

Actors Involved: Player

Aim and Response: Player wants to change his/her account so he/she can logout from there. Also he/she wants to turn on or turn off music so he/she can change music option.

Pre-condition: Player has to be in main menu or playgame screen.

Post-condition: Player changed music option or logged out.

Entry Condition: Player selects “Settings” button from main menu screen or playgame screen.

Exit Condition : Player selects “Back” button.

Event Flow :

1. Player clicks “Settings”button.
2. Settings page opens.

Flow A :

3A: Player does not want to change any settings so he/she exits page.

Flow B:

3B: Player turns off or on music sound via switch button.

Flow C:

3C: Player logouts in order to login another or same account.

4C: Login page opens.

4.1.5 Credit

Use case 5: Credit

Actors Involved: Player

Aim and Response: If player wants to see developers of software, version of the application, update date, these are demonstrated in credit page.

Pre-condition: Player has to be in main menu. Then, click “Credit” button.

Post-condition: Player backs to the main menu.

Entry Condition: Player has to login before going in to main menu.

Exit Condition: Player pushes the back button.

4.1.6 Info&Help

Use case 6: Info&Help

Actors Involved: Player

Aim and Response: If player wants to learn how to play, or features of buttons and tricks.

Pre-condition: Player has to be in main menu. Then, click “Info&Help” button.

Post-condition: Player backs to the main menu.

Entry Condition: Player has to login before going in to main menu.

Exit Condition: Player pushes the back button.

4.1.7 High Score

Use case 7: High Score

Actors Involved: Player

Aim and Response: Player should be able to see other users scores so that competition may increase. Top 20 users are going to be displayed.

Pre-condition: Player has to be in main menu. Then, click “High Score” button.

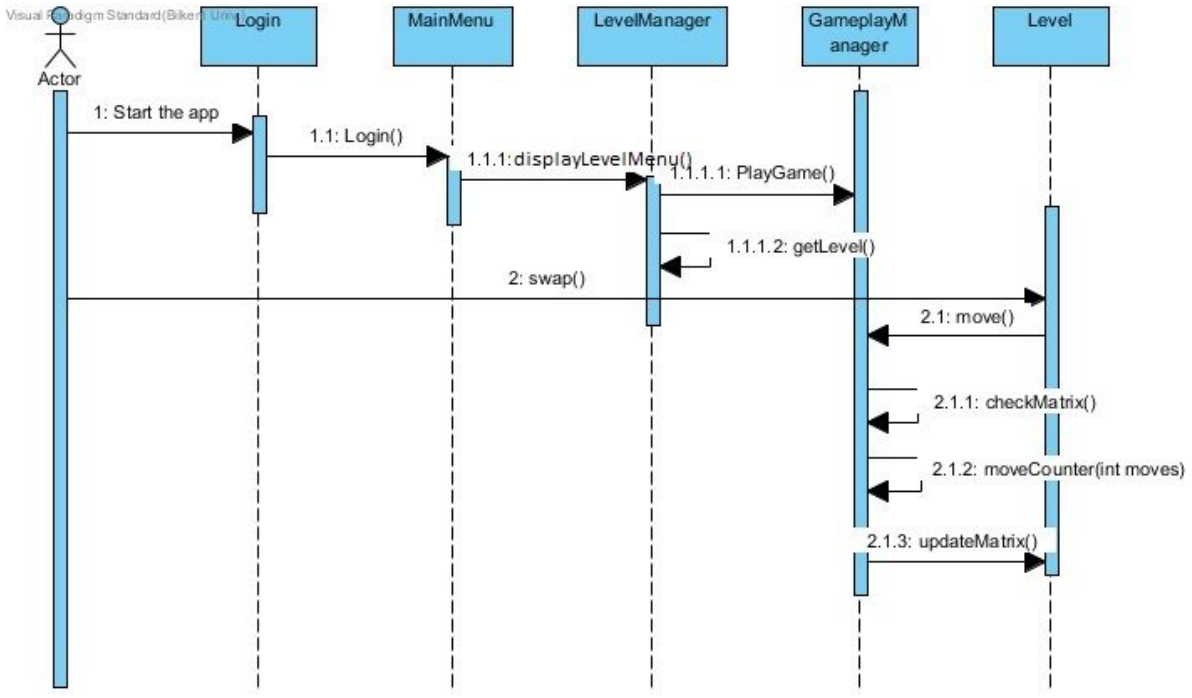
Post-condition: Player backs to the main menu.

Entry Condition: Player has to login before going in to main menu.

Exit Condition: Player pushes the back button.

4.2 Dynamic Model

4.2.1 Start Game

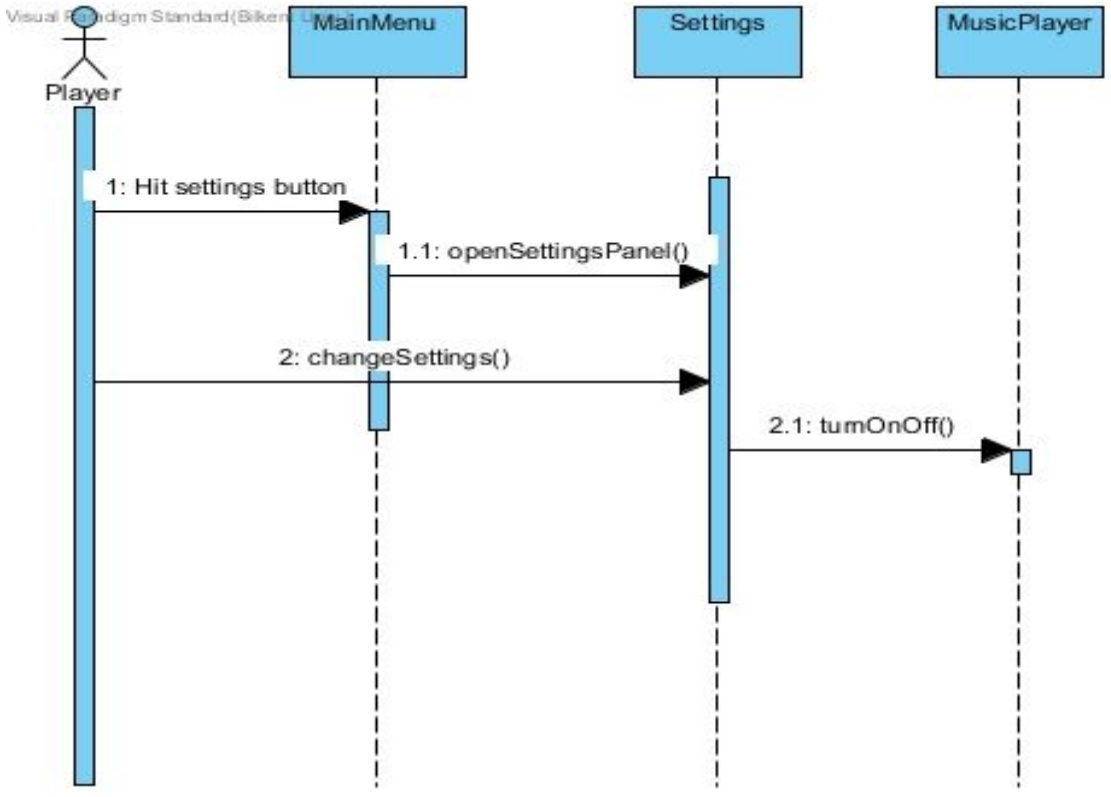


(figure 6)

Event Flow:

After player logged in, main menu is opened. Play Game, Settings, Info & Help , High Score, Credit and Exit Buttons are in main menu window. When the player clicks on the Play Game button, level window is opened. In this window, some levels are enable some levels are not enable. This is because, if the player want to play a level, the player must collect the enough points of previous level. When the player clicks on an enable level, the game starts. After starting, the player try to order at least 3 same books one after another by swapping. Moreover, there is a limited swap number according to the level. moveCounter(int moves) counts the number of moves player makes and compares it to move limit which level has. Then the books are disappeared and the player gains point. After disappearing, other books which are above fill the spaces.The player collect sufficient point to access a top level.The game continues in this way.

4.2.2 Settings



(figure 7)

Event Flow:

When the player clicks on the setting button in the main menu, a window which includes a switch opens. This switch is need for controlling the music of the game. On the other hand, the settings window can be opened while the gamer is playing the game.

4.2.3 View Help

Event Flow:

Player wants to get information about the game before playing the game or while playing game.

Player clicks the “Info & Help” button and navigates to “Info & Help” page. Information about courses and boosters and instructors in real life is displayed.

4.2.4 View Credits

Event Flow:

Player wants to get information about developers so he/she clicks the “Credits” button. The system displays personal background and contact information about developers.

4.2.5 View High Scores

Event Flow:

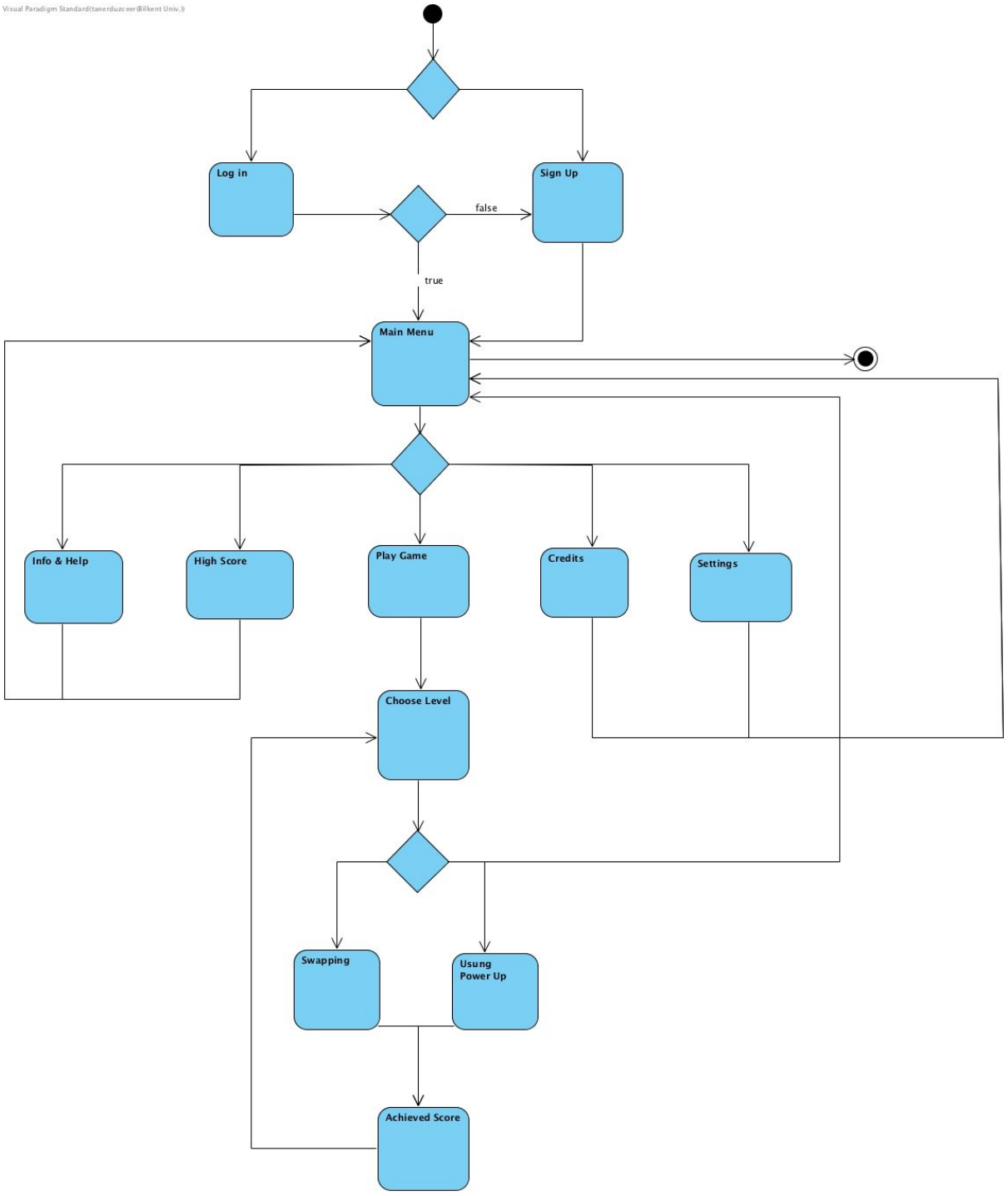
Player wants to see his/her rank among others and also first twenty highscored people so he/she clicks “High Scores” button. The system gets the high scores from database and displays them in “High Scores” page.

4.2.6 Exit Game

Event Flow:

Player wants to exit game so he/she clicks “Exit” button. The game and every processes of game get terminated.

4.3 Activity Diagram



4.4 Class Diagrams

figure 4.4 Class Diagram

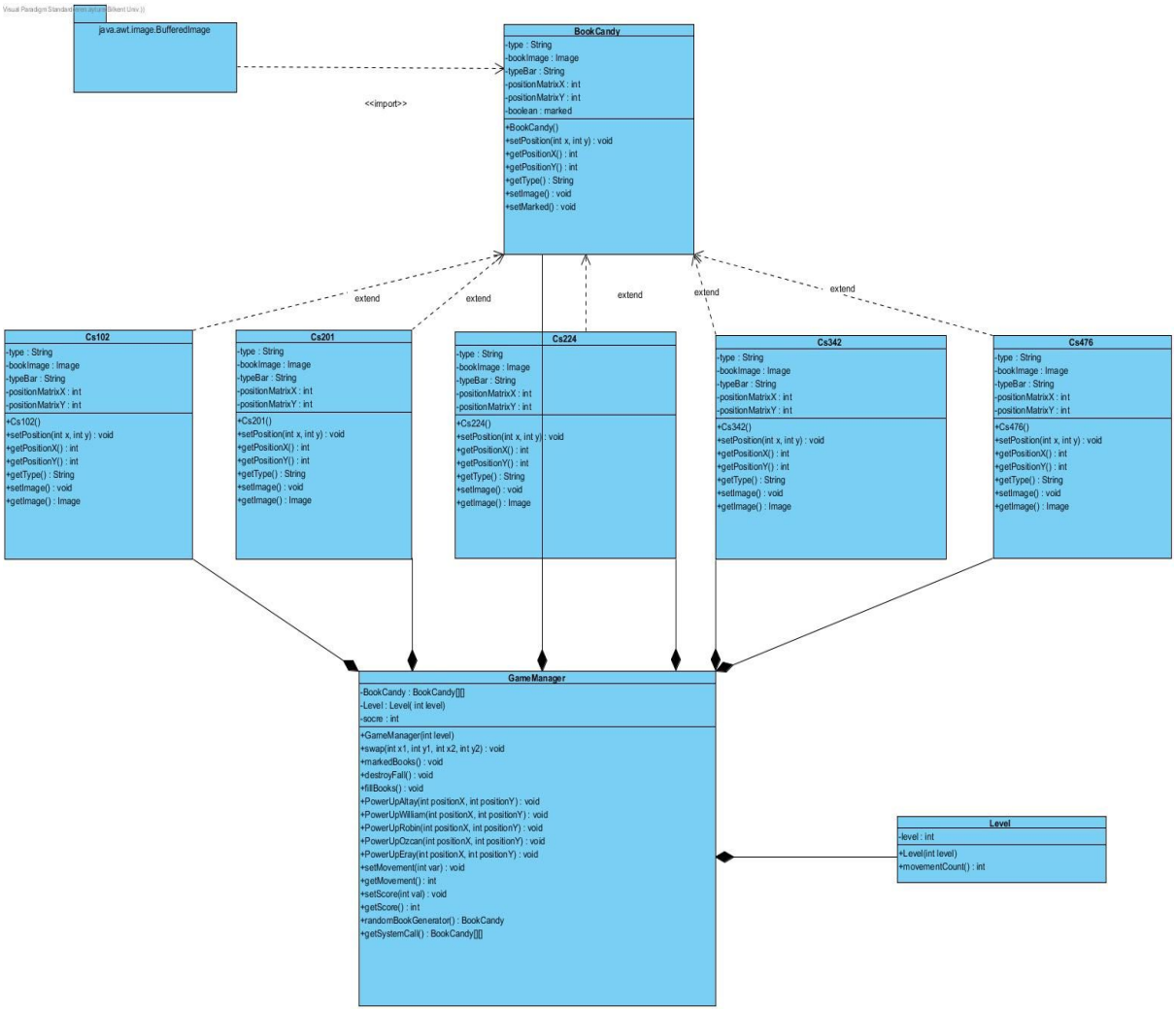


figure 4.5 Game Logic

5. Navigational Path

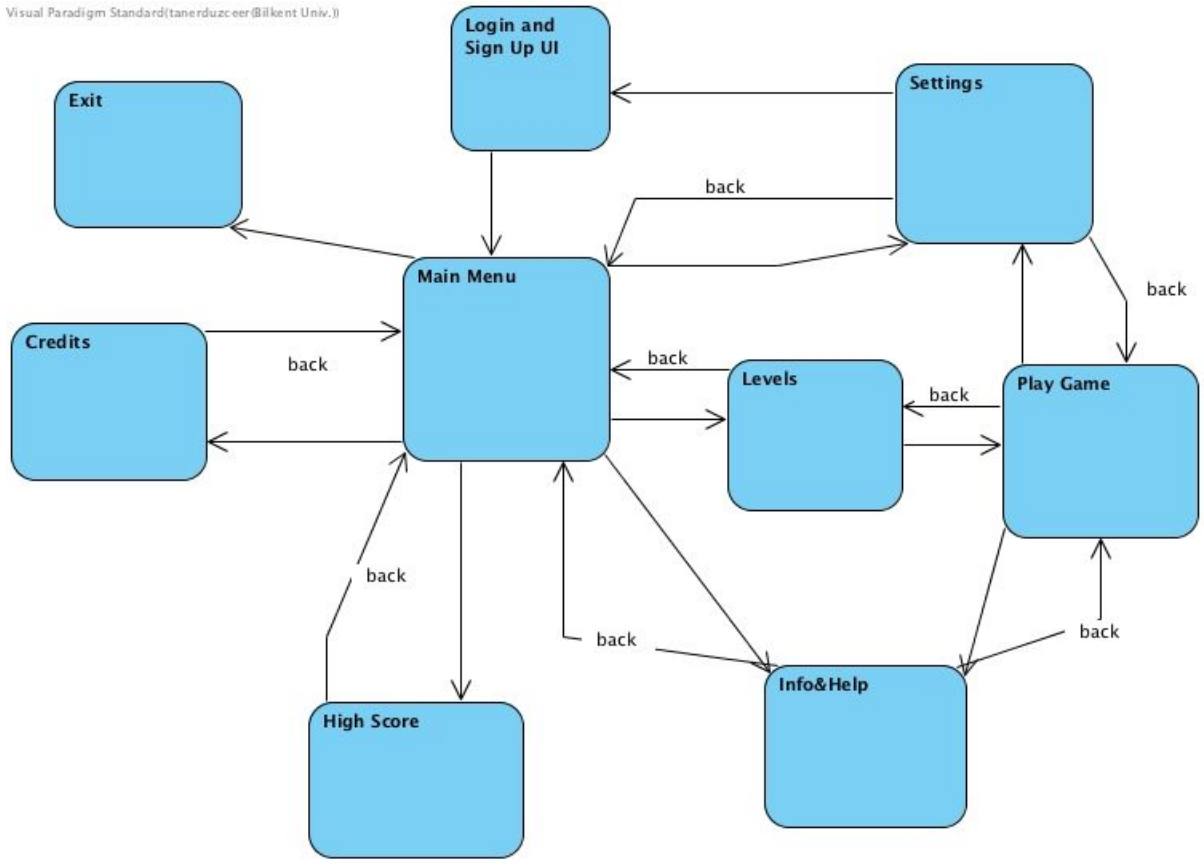
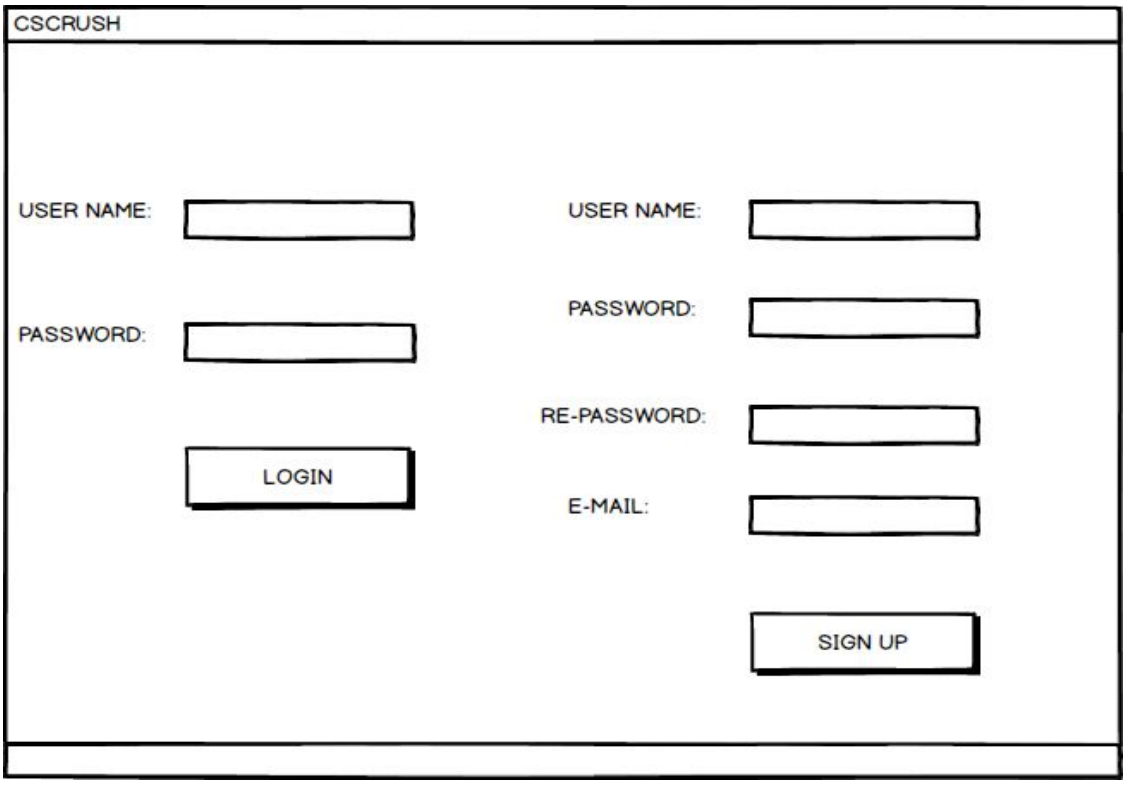
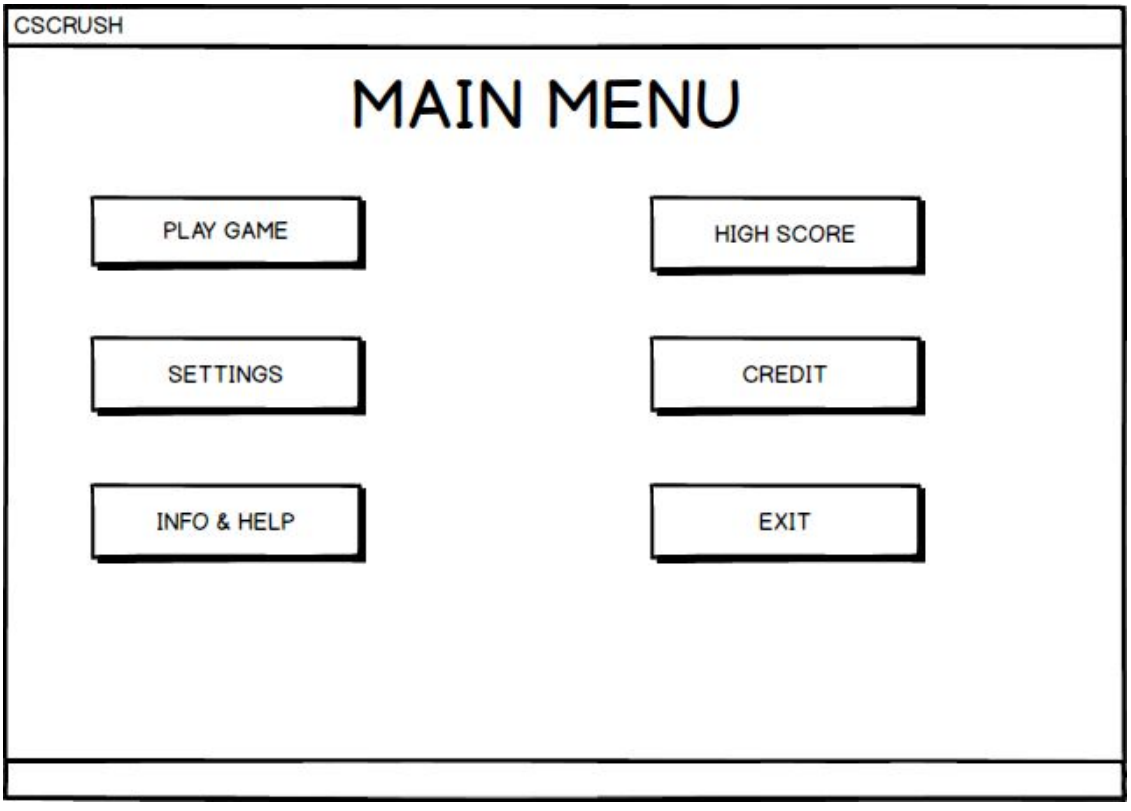
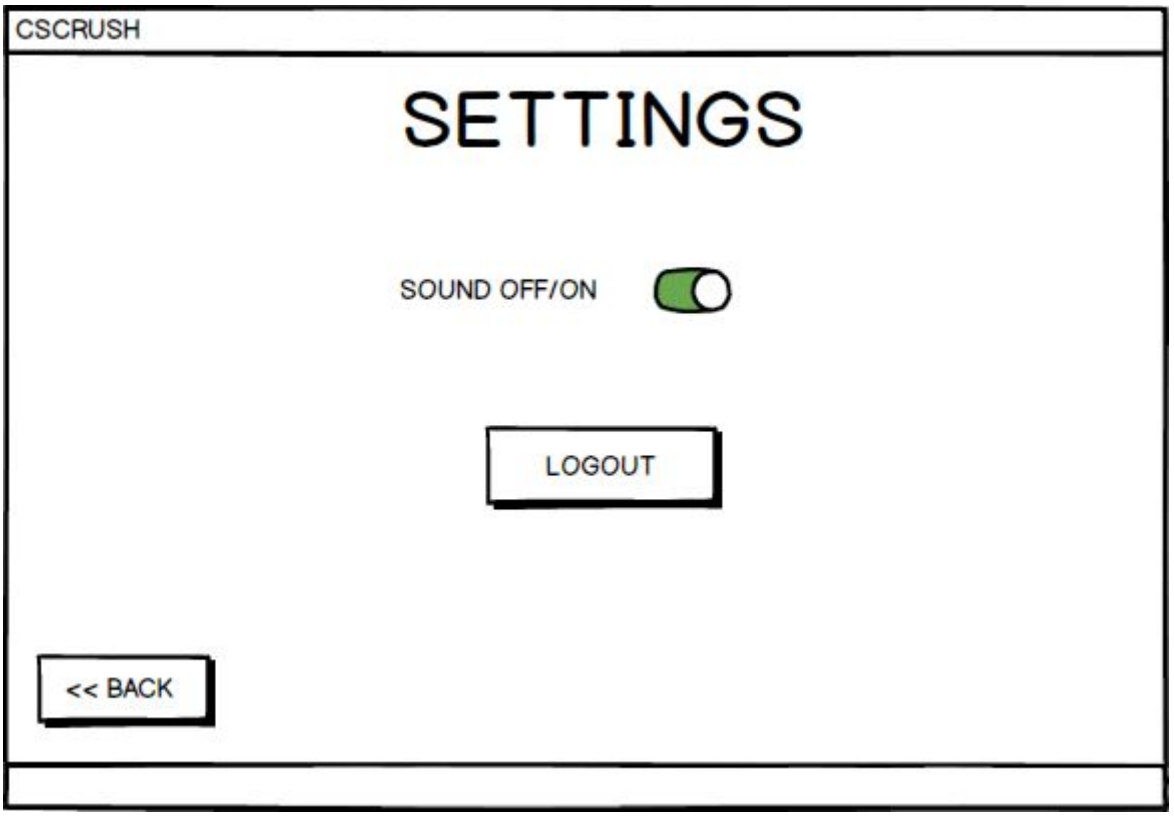


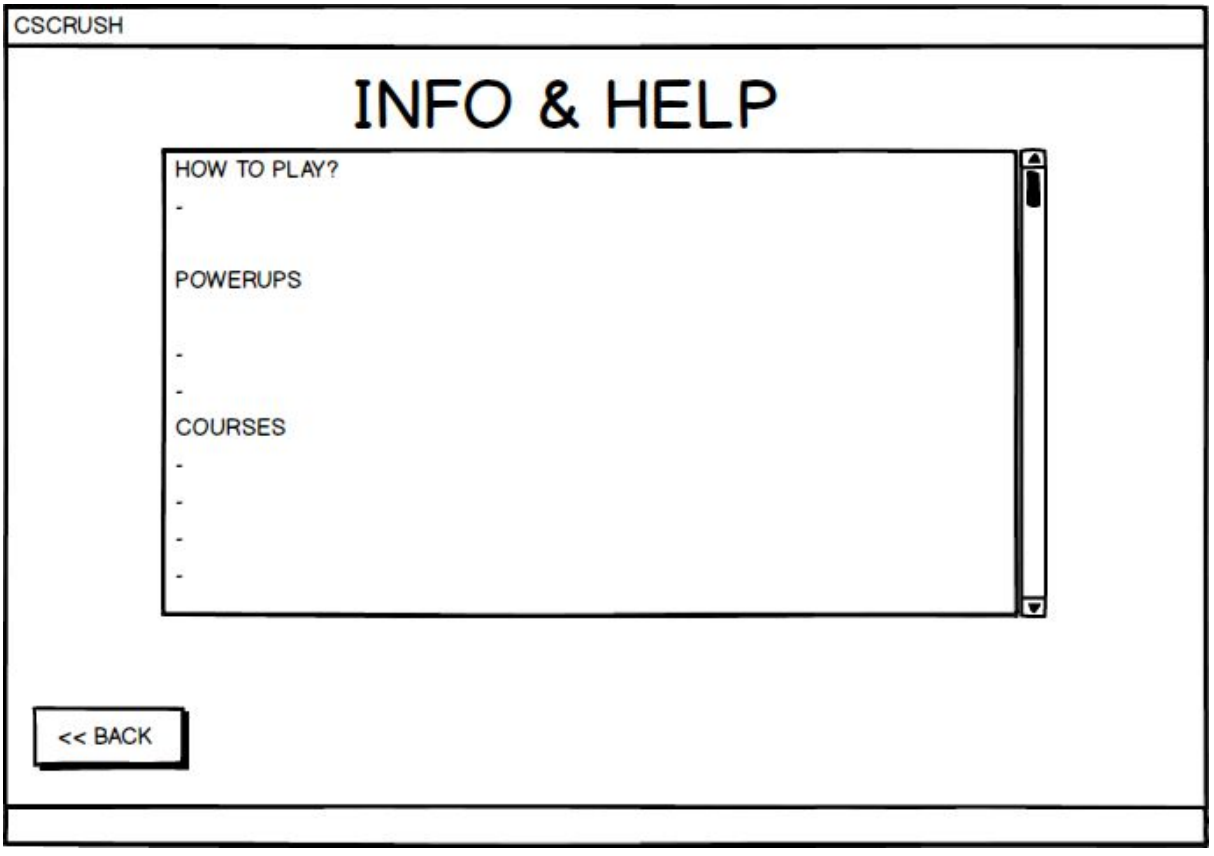
figure 5 Navigation Path Diagram

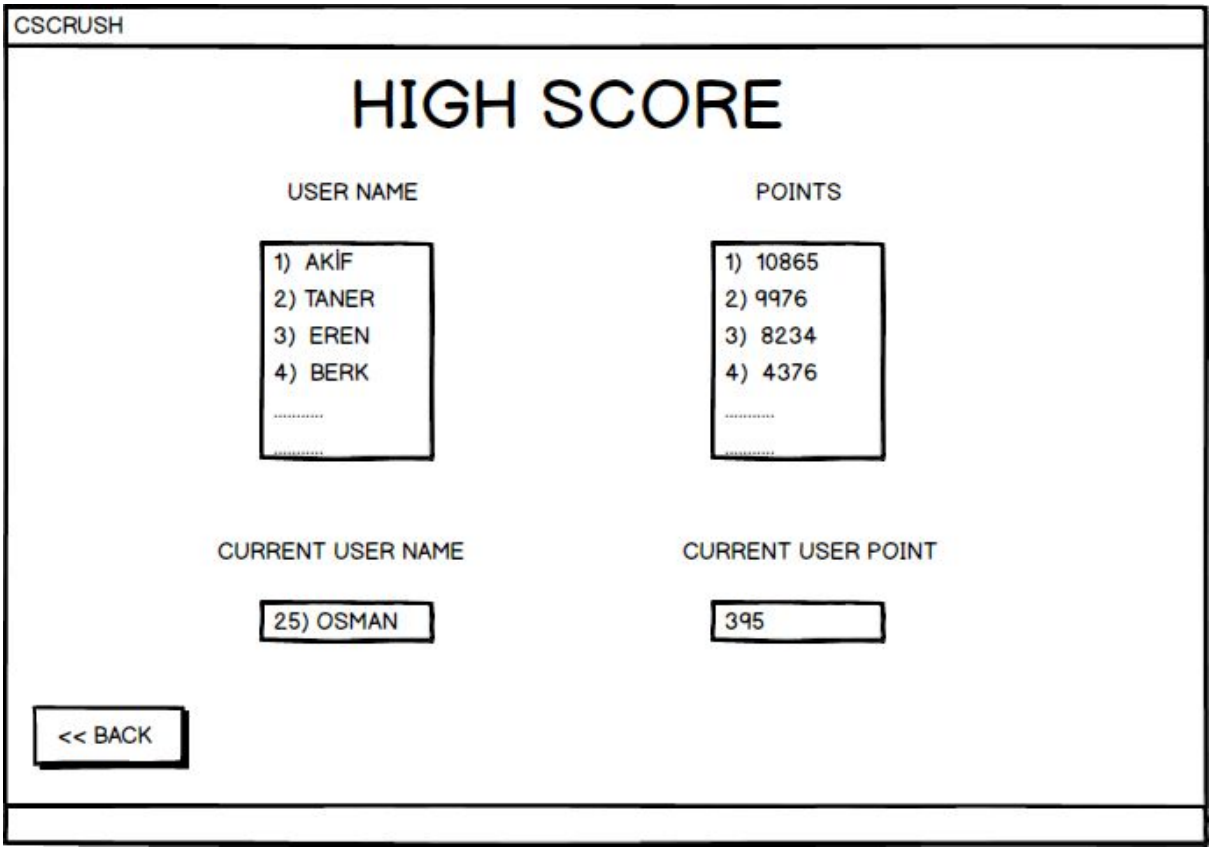
6. Screen Mock-Ups

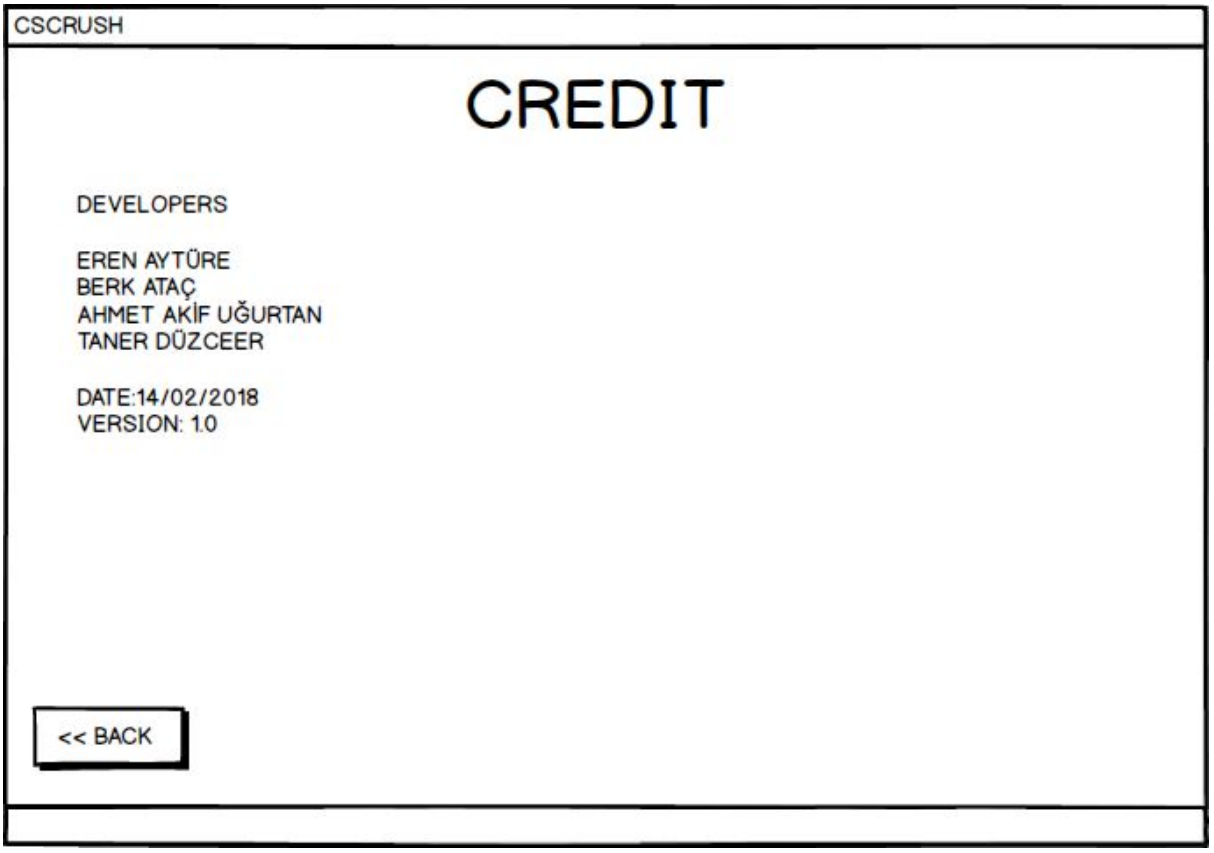


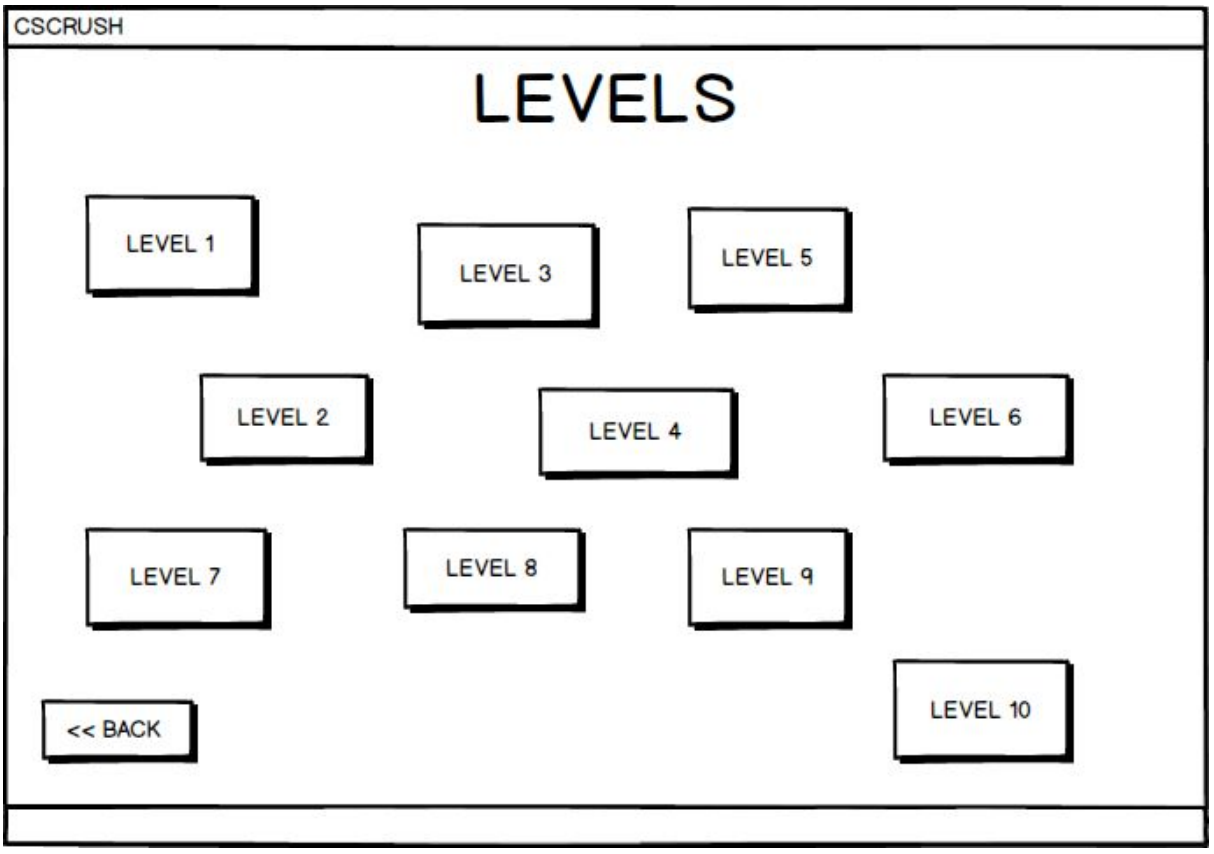


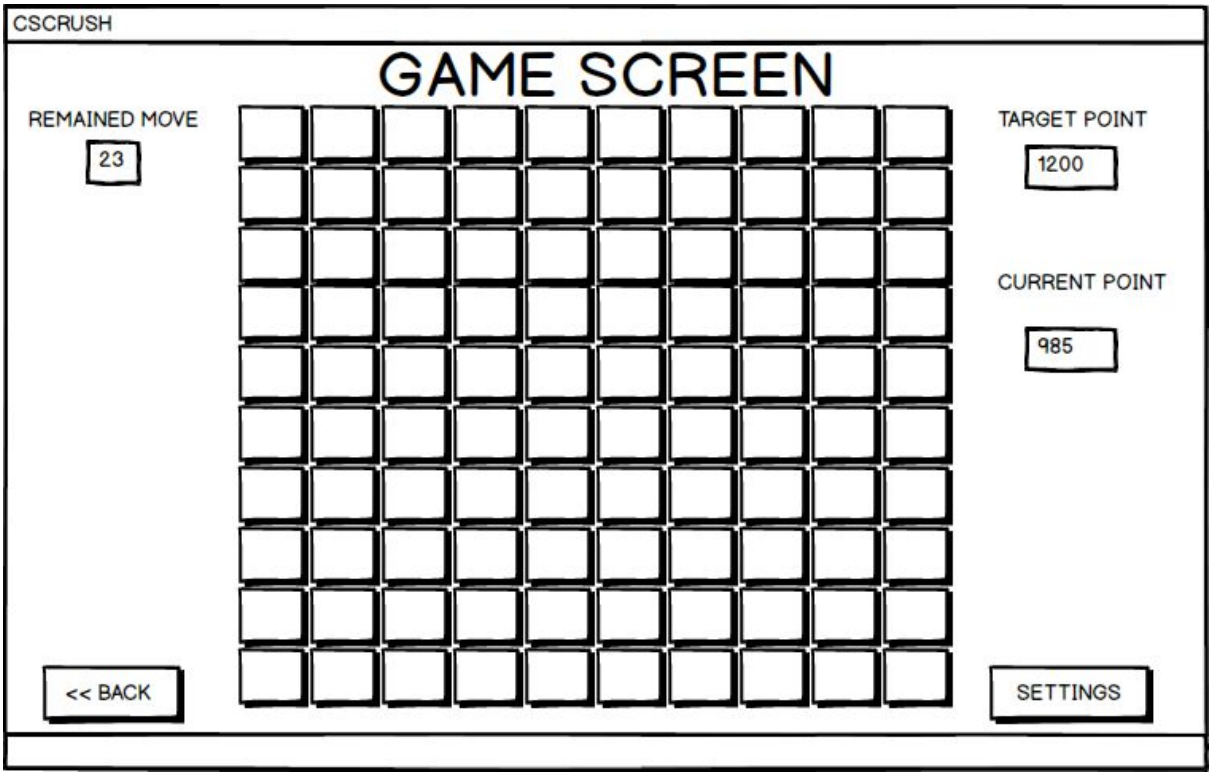

CSCRUSH
SETTINGS
SOUND OFF/ON
LOGOUT
<< BACK


CSCRUSH
INFO & HELP
HOW TO PLAY?
-
POWERUPS
-
-
COURSES
-
-
-
-
<< BACK


CSCRUSH
HIGH SCORE
USER NAME
POINTS
1) AKİF
2) TANER
3) EREN
4) BERK
.....
.....
1) 10865
2) 9976
3) 8234
4) 4376
.....
.....
CURRENT USER NAME
CURRENT USER POINT
25) OSMAN
395
<< BACK


CSCRUSH
CREDIT
DEVELOPERS
EREN AYTÜRE
BERK ATAÇ
AHMET AKİF UĞURTAN
TANER DÜZCEER
DATE:14/02/2018
VERSION: 1.0
<< BACK


CSCRUSH
LEVELS
LEVEL 1
LEVEL 3
LEVEL 5
LEVEL 2
LEVEL 4
LEVEL 6
LEVEL 7
LEVEL 8
LEVEL 9
LEVEL 10
<< BACK


CSCRUSH
GAME SCREEN
REMAINED MOVE
23
TARGET POINT
1200
CURRENT POINT
985
<< BACK
SETTINGS

7. Design Goals

The design goals are important to meet the qualification of the application. So, we paid our attention to functional and nonfunctional components of the system. Important design goals are listed below

Performance Criteria:

Response Time and Throughput

The game is designed to satisfy response time. Namely, when the user click on a book icon and another book icon, the game should perform swap animation smoothly no more than 1 secdon delay. The application is going to be responsive and it will work with high performance.

Memory

This application does not require no more than 1 GB memory. Game variables will be kept in memory.

Dependability Criteria:

Reliability

The game will be designed with the purpose that it will not produce any bugs. We will control all of the bugs during and at the end of development process. It will run on any desktop platform using Java 8 and above.

Cost Criteria:

Development Cost

CSCrush applicaiton is not a commercial product. The external components of the system like images, icons, vs. obtained from the existing free resources. This application can be installed and used freely.

Maintenance Criteria:

Modifiability

Adding some new features is important for the future of the game. Moreover, the users attention can be gained by adding new features. These features can be new power ups, new type of book icons vs. On the other hand, the Appleton can be developed according to the user's suggestions.

Portability

Our application is going to be developed by using java. Java works on portable Java Virtual Machine(JVM). This is because, this platform can work on different machines. Therefore, our application is portable.

Readability

The designation of application is simple. In that sense, the code of the application can be understandable. To improve the understandability, we will add extra comments.

End User Criteria:

Usability

The simplicity of the usage of the game is important design goal because the easiness of usage provide us user friendliness and the user's attraction. The simplicity of the game is related to user interface. Swapping the icons can be done by clicking on the mouse. Power ups will be on the screen. Namely, the user can reach every components of the game easily. Moreover, the simplicity of the user interface is related to the simplicity of the design of the game.

Design Goal Trade Offs:

Server Reponse Time vs. Modifiability

Game icon pictures will be held in game directory, they will not be taken from server in order not to engage the server. They fill be fixed images, unless an update comes, icons will not be changeable.

Definitions, acronyms and abbreviations

Cross-platform

It means that a program is able to run in different platforms such as Windows, Linux, MAC OS properly in a same way.

JRE

Java Runtime Environment in short JRE is a set of software tools for Java applications and Java programming.

FPS

Frames Per Second in short FPS is the number of different frames in one second. Animations are usually 60 FPS since it seems better and nicer to human eye.

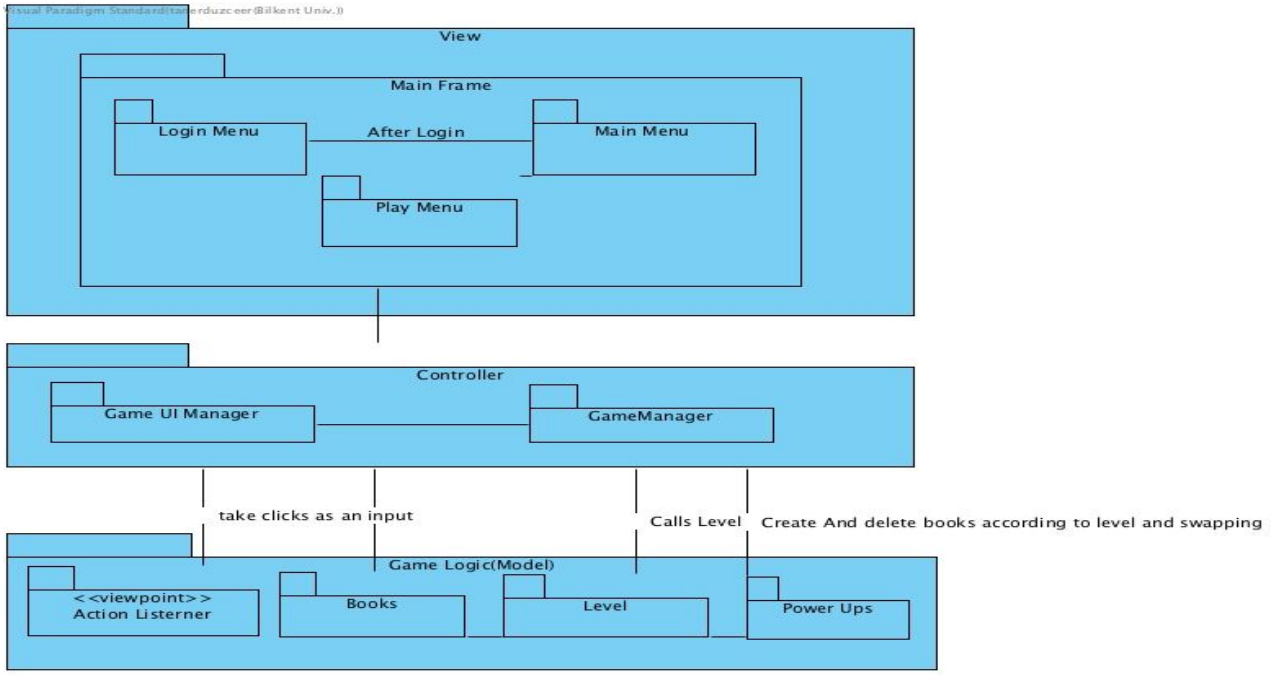
8. Software Architecture

This section will show the software architecture of the system. The Main System will decompose into subsystems in a way of Model View Controller architectural style. Thus, the system will be more controllable.

8.1 Subsystem Decomposition

By dividing Main system into several subsystems, the program will be more modifiable and extendible. There are three subsystems named Model such as game logic algorithms, View such as user interface objects, and Controller such as game managers. This flow shows the logic from user interface to game logic via controllers. Subsystems have their entities with similar functionalities grouped as views or logic or

controllers. Through subsystems and workflow, the program will meet the non-functional requirements. If there needs any modification to improve the program efficiently, this architectural path will allow us to modify program without contradictions.



UI and Gameplay Layer interactions (Detailed)

8.2 Architectural Styles

8.2.1 View (User Interface)

User interface components include in this subsystem. Interaction with buttons, switches, text boxes etc. will be managed in this subsystem. The CSCrush game will have only one frame which is MainFrame. This frame will be used to show the game to user via changing panels to change context.

8.2.2 Controller

This component arranges the interaction between the user interface and game logic via GameUIManager and GameManager. Controller will get the actions

from View and will send these actions to the Model so that data will be modified by calculations. Then, It will get the data from game logic Model and will send that data to appropriate View.

8.2.3 Model (Game Logic)

Game logic component determines the last situation of the game by user inputs. For instance, the user wants to swap books, this component executes the necessary calculations. Moreover, while playing the game, this components gives a decision about which books are disappeared or which books come in. Shortly, this component models the game on the way of user' inputs.

8.3 Hardware/Software Mapping

Since the game will be written in JAVA, we will use JRE (Java Runtime Environment). Thanks to JRE, it will be able to run at any device which has JRE. Game will run on RAM and one mouse is enough to play active game but a keyboard is necessary to write username and password while login or signup. In order to check user properties or to update new high score or to display high score table, the program will communicate with database which will run on another computer serving as a JAVA server.

8.4 Persistent Data Management

We will use relational database system via MySQL database manager. There will be user entity, highscore entity, level entity and relations between them.

Users have to login every time to play game so that after sign up, user properties will be stored in a database. While login, the program will communicate with the database in order to check user credentials and to get user properties such as high score of user for each level.

User is able to display high scores by opening the high score interface via the button in main menu, the system must get best high scores and rank of user from database since user is challenging with other users and there must be synchronization.

After each breaking the record of high score of a level, the system will update the high score of this level in database for current user.

8.5 Access Control and Security

In order to play game, users have to have game .jar file and also internet connection. Nowadays, connecting the internet is not a big issue so we thought this will not be a problem and thanks to internet users will able to challenge each other by trying to get more points than others or themselves. In order to separate users from each other, each user has to have username and password. Also an email address is necessary for verifying user to prevent bot users.

The system will not share any user credentials with public to ensure security and privacy. To fulfill this aim, we will use AES symmetric key encryption to preserve data encrypted. While communication between client and server, to ensure safety we will use RSA public key encryption so the data will be send encrypted by encrypting with receiver's public key and only the receiver will be able to read it by decrypting with his/her private key.

8.6 Global Software Control

Event-driven Control

In our program, anything will be updated after clicking buttons or swiping images in game screen. Those button or mouse events will be controlled thanks to listeners in JAVA. GameUIManager will get responses from other UI classes and will decide what to do and GameManager will get changes from GameUIManager and it will update background data.

8.7 Boundary Conditions

Start-up

JAVA 8 or higher must be installed at device since game runs in JAVA. User has to have internet connection and game .jar file to play game.

Shutdown

User can terminate game by clicking the “Exit” button in main menu.

Exception Handling

User will be notified in case of any crashes such as not loading an image, not connecting the database, not having an internet connection.

9. Subsystem Services

9.1 Design Pattern

Behavioral Patterns

Facade

We will use Facade design pattern to create an interface for book and booster classes in order to display them in GameplayPanel class since we have a lot of types of books and boosters. GameManager will be the Facade class which calls all other update methods. By doing this, we delegated the calls from GameplayPanel to game logic classes. This provides us more flexible working areas.

MVC

We will use Model View Controller design pattern to make changes easily later. Model part will include only the game logic and data. View part will include only view objects for user interface. Controller part will set the communication between Model and View.

9.2 Designs

9.2.1 User Interface Subsystem

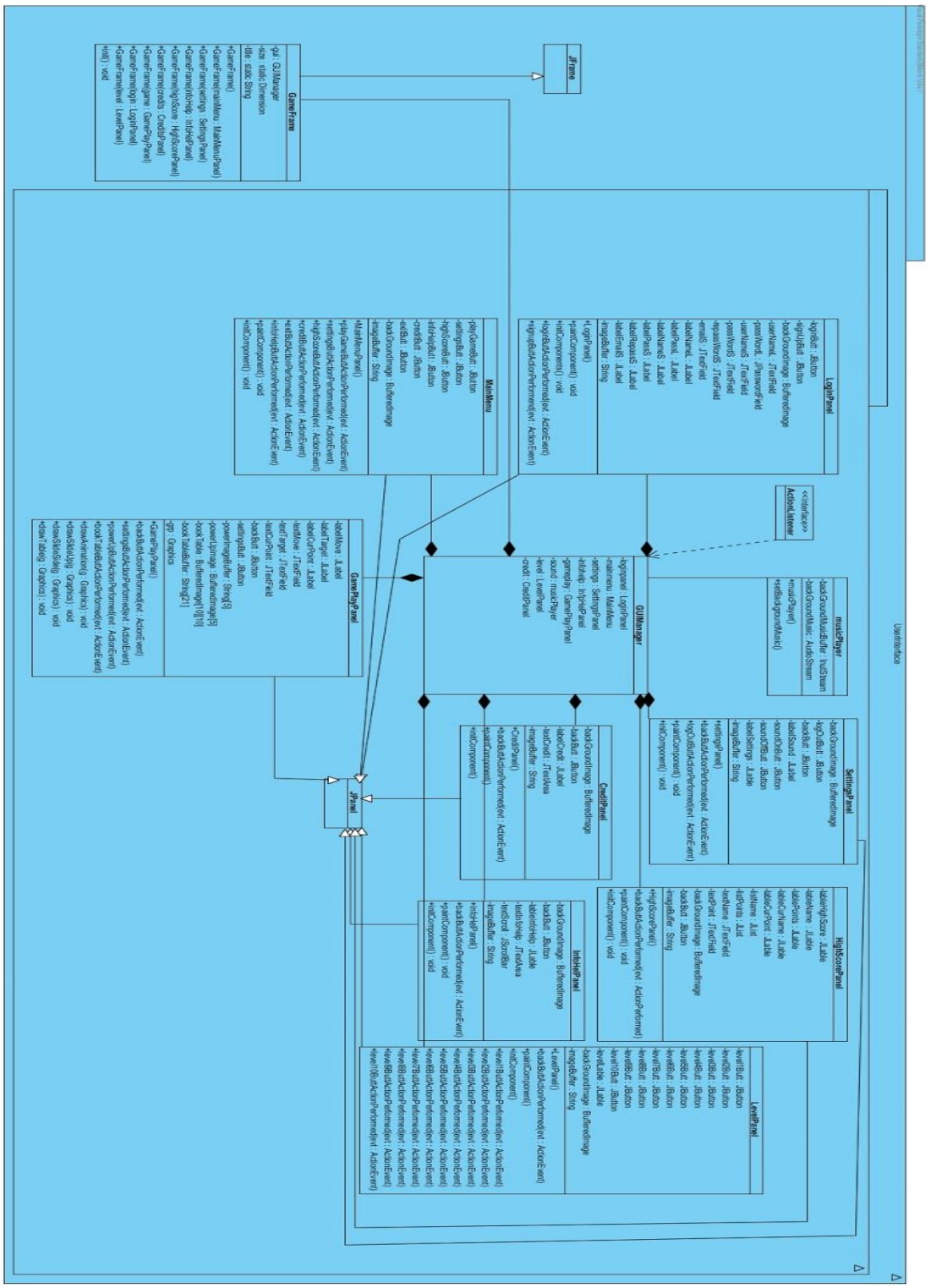
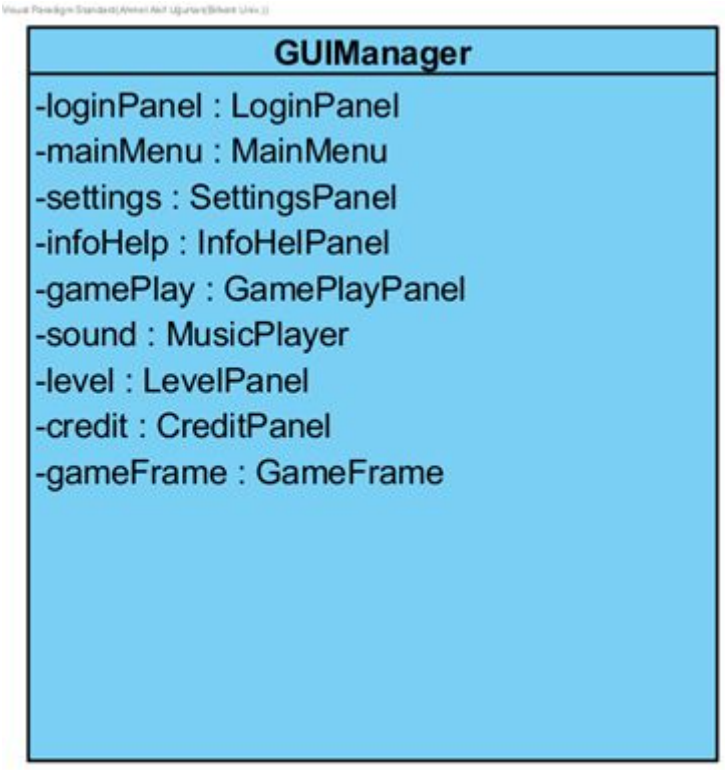


figure 9.2.1

GUIManager Class



Attributes

private LoginPanel loginPanel : Instance of LoginPanel class which is created and managed by GUIManager class to set main panel of gameFrame as this class if user navigates to login page.

private MainMenu mainMenu : Instance of MainMenu class which is created and managed by GUIManager class to set main panel of gameFrame as this class if user navigates to main menu page.

private Settings settings : Instance of Settings class which is created and managed by GUIManager class to set main panel of gameFrame as this class if user navigates to settings page.

private InfoHelPanel infoHelp : Instance of InfoHelPanel class which is created and managed by GUIManager class to set main panel of gameFrame as this class if user navigates to information and help page.

private GamePlayPanel gamePlay : Instance of GamePlayPanel class which is created and managed by GUIManager class to set main panel of gameFrame as this class if user navigates to game playing page.

private MusicPlayer sound : Instance of MusicPlayer class which is created

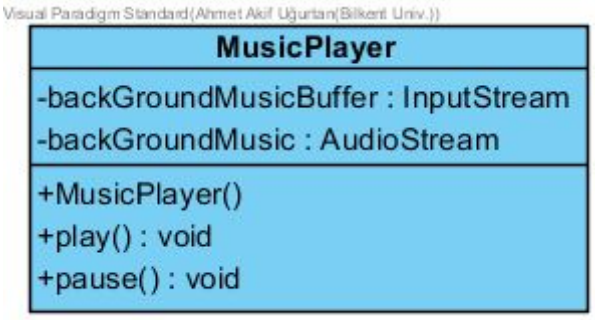
and managed by GUIManager class to play music on background in every page.

private LevelPanel level : Instance of LevelPanel class which is created and managed by GUIManager class to set main panel of gameFrame as this class if user navigates to level page.

private CreditPanel credit : Instance of CreditPanel class which is created and managed by GUIManager class to set main panel of gameFrame as this class if user navigates to credits page.

private GameFrame gameFrame : Instance of GameFrame class which is created and managed by GUIManager class to display panels during navigation between them.

MusicPlayer Class



Attributes

private InputStream backGroundMusicBuffer : This attribute saves an InputStream in order to be ready to play music.

private AudioStream backFroundMusic : After input stream is received this instance uses AudioStream class to start playing music.

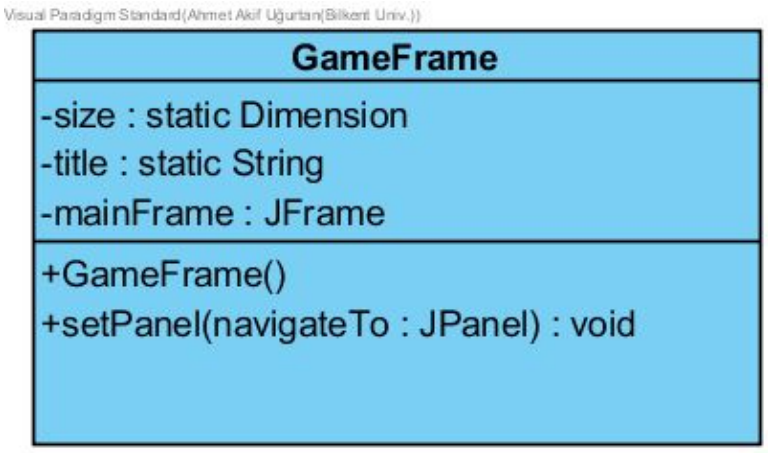
Methods

public MusicPlayer() : Constructor for MusicPlayer class to initialize an instance of MusicPlayer.

public void play() : A method to start playing music.

public void pause() : A method to pause music.

GameFrame Class



Attributes

private JFrame mainFrame : Only Frame object in whole application to display panels.

private static Dimension size : A variable to save default frame size.

private static String title : Name of the game to be displayed on top of the frame.

Methods

public GameFrame() : Constructor for GameFrame class to initialize instance and create a JFrame.

public void setPanel(JPanel navigateTo) : This method changes the current panel object with navigateTo object so that the content in mainFrame changes and user navigates to desired page.

LoginPanel Class

Attributes

private JButton loginButt : Instance of JButton class to listen whether user wants to login or not.

private JButton signUpButt : Instance of JButton class to listen whether user wants to signup or not.

private JTextField userNameL : Instance of JTextField to get username from user when he/she clicks the loginButt button.

private JPasswordField passwordL : Instance of JPasswordField to get password from user when he/she clicks the loginButt button.

private JTextField userNameS : Instance of JTextField to get username from user when he/she clicks the singUpButt button.

private JPasswordField passwordS : Instance of JPasswordField to get password from user when he/she clicks the signUpButt button.

private JPasswordField repasswordS : Instance of JPasswordField to get password again to check with passwordS from user when he/she clicks the signUpButt button.

private JTextField emailS : Instance of JTextField to get email from user when he/she clicks the singUpButt button.

private JLabel labelNameL : Instance of JLabel to display “Username” label before userNameL text field to make user comprehend what he/she should enter in the text field.

private JLabel labelPassL : Instance of JLabel to display “Password” label before passwordL text field to make user comprehend what he/she should enter in the text field.

private JLabel labelNameS : Instance of JLabel to display “Username” label before userNameS text field to make user comprehend what he/she should enter in the text field.

private JLabel labelPassS : Instance of JLabel to display “Password” label before passwordS text field to make user comprehend what he/she should enter in the text field.

private JLabel labelRepassS : Instance of JLabel to display “Repassword” label before repasswordS text field to make user comprehend what he/she should enter in the text field.

private JLabel labelEmailS : Instance of JLabel to display “Email” label before emailS text field to make user comprehend what he/she should enter in the text field.

private BufferedImage backgroundImage : An instance to get and save background image then display in the screen.

private String imageBuffer : Direction of background image file.

Methods

public LoginPanel() : Constructor for LoginPanel class to initialize attributes.

public void paintComponent() : Overriding javax.swing.JPanel method to

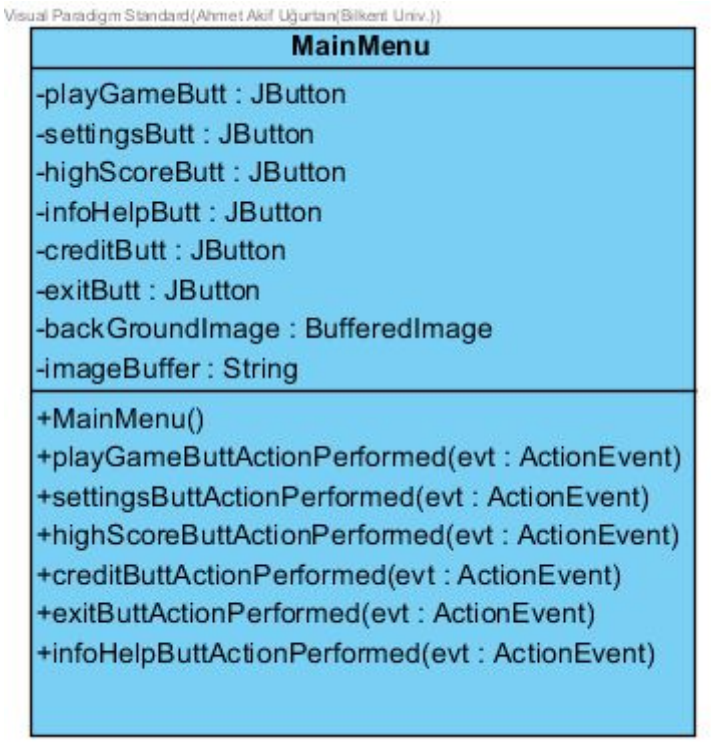
paint objects over and over again.

public void initComponents() : This method initialize attributes by being called from constructor.

public void loginButtActionPerformed(ActionEvent evt) : Actionlistener method for loginButt button in order to post a request to server and login.

public void signupButtActionPerformed(ActionEvent evt) : Actionlistener method for signUpButt button in order to post a request to server and sign up.

MainMenu Class



Attributes

private JButton playGameButt : A JButton instance to navigate Level page then playGame screen.

private JButton settingsButt : A JButton instance to navigate Settings page.

private JButton highScoreButt : A JButton instance to navigate High Score page.

private JButton infoHelpButt : A JButton instance to navigate Information and Help page.

private JButton creditButt : A JButton instance to navigate Credits page.

private JButton exitButt : A JButton instance to Exit and terminate game.

private BufferedImage backGroundImage : An instance o BufferedImage to get image and later to display background image.

private String imageBuffer : The direction of image file for background.

Methods

public MainMenu() : Constructor for MainMenu class to initialize objects.

public void playGameButtActionPerformed(ActionEvent evt) : Button listener for playGameButt.

public void settingsButtActionPerformed(ActionEvent evt) : Button listener for settingsButt.

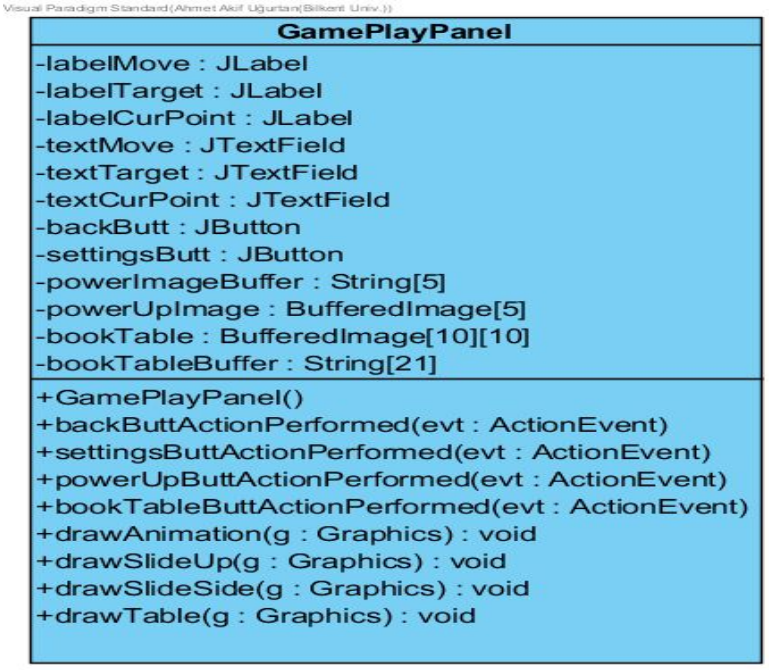
public void highScoreButtActionPerformed(ActionEvent evt) : Button listener for highScoreButt.

public void infoHelpButtActionPerformed(ActionEvent evt) : Button listener for infoHelpButt.

public void creditButtActionPerformed(ActionEvent evt) : Button listener for creditButt.

public void exitButtActionPerformed(ActionEvent evt) : Button listener for exitButt.

GamePlayPanel Class



Attributes

private JLabel labelMove : Instance of JLabel to display “Remained Move”

label before textMove text field to make user comprehend what that field means.

private JLabel labelTarget : Instance of JLabel to display “Target Point” label before textTarget text field to make user comprehend what that field means.

private JLabel labelCurPoint : Instance of JLabel to display “Current Point” label before textCurPoint text field to make user comprehend what that field means.

private JTextField textMove : Instance of JTextFied to display number of remained movements.

private JTextField textTarget : Instance of JTextFied to display target point.

private JTextField textCurPoint : Instance of JTextFied to display current achieved point.

private JButton backButt : A JButton instance to navigate back to the Level page.

private JButton settingsButt : A JButton instance to navigate Settings page.

private String powerImageBuffer[5] : Directions of images of power up objects.

private BufferedImage powerUpImage[5] : An instance array to get and save images of powerup objets.

private BufferedImage bookTable[10][10] : An instance array to get and save images of CSCrush objets. This array is the main game play table since all CSCrush objects and sliding, crushing animations will be represented here.

private String bookTableBuffer[21] : Directions of 21 images in CSCrush objects such as books, horizontal lined books, vertical lined books etc.

Methods

public GamePlayPanel() : Constructor for GamePlayPanel Class to initialize attributes.

public void backButtActionPerformed(ActionEvent evt) : Action listener for backButt.

public void settingsButtActionPerformed(ActionEvent evt) : Action listener for settingsButt.

public void powerUpButtActionPerformed(ActionEvent evt) : Action listener for powerUp images.

public void bookTableButtActionPerformed(ActionEvent evt) : Action listener for images on the gameplay table.

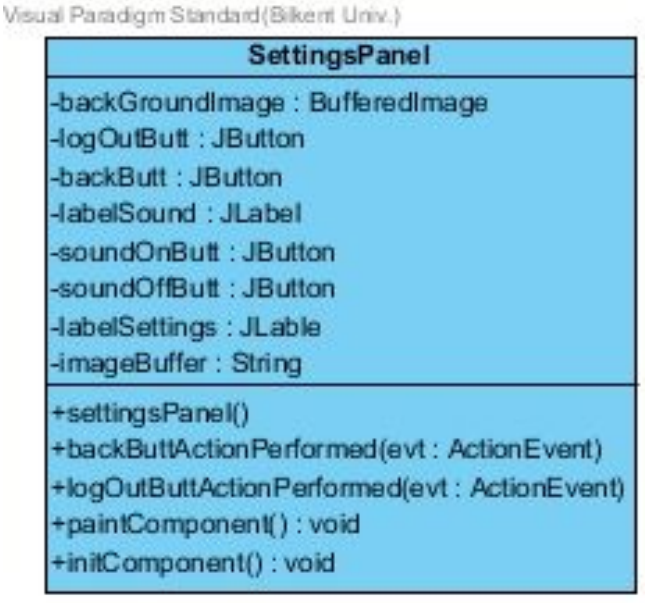
public void drawAnimation(Graphics g) : This method draws animations while crushing CS books.

public void drawSlideUp(Graphics g) : This method shows the animation while sliding horizontally.

public void drawSlideSide(Graphics g) : This method shows the animation while sliding vertically..

public void drawTable(Graphics g) : This method draws and redraws table images.

SettingsPanel Class



Attributes

BufferedImage backGroundImage: The image to display in the background of SettingsPanel.

String imageBuffer: The directory to read the image to display in the SettingsPanel.

private javax.swing.JButton backButt: In settings, a JButton instance is called and it's modified to a button with "to back page" functionality.

private javax.swing.JButton logOutButt: : In settings, a JButton instance is called and it's modified to a button with "log user out" functionality.

private javax.swing.JButton soundOnButt: : In settings, a JButton instance is called and it's modified to a button with "turn on sound"functionality.

private javax.swing.JButton soundOnButt: : In settings, a JButton instance is called and it's modified to a button with "turn off sound"functionality.

private javax.swing.JLable labelSound: A JLable instance for indicating sound options.

private javax.swing.JLable labelSettigs: A JLable instance for title of the settings page.

Methods

public SettingsPanel() : Constructor for SettingsPanel class to initialize objects.

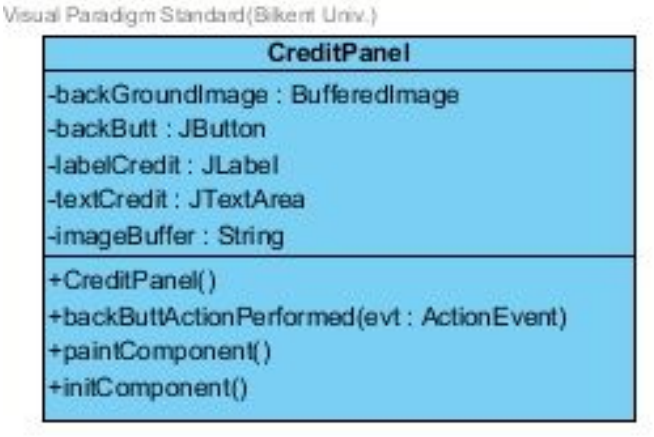
public void backButtActionPerformed(ActionEvent evt) : Action listener for backButt.

public void logOutButtActionPerformed(ActionEvent evt) : Action listener for logOutButt.

paintComponent () : To draw JPanel

initComponent (): To initialize components.

Credit Panel Class



Attributes

BufferedImage backGroundImage: The image to display in the background of CreditsPanel.

String imageBuffer: Address of image to be used is defined with this String attribute.

private javax.swing.JTextArea textCredit: This is a text area for developer names to appear.

private javax.swing.JButton backButt: This is a Button displayed in Credits panel which will be used to go back to main menu.

private javax.swing.JLabel labelCredit: This is the title string.

Methods

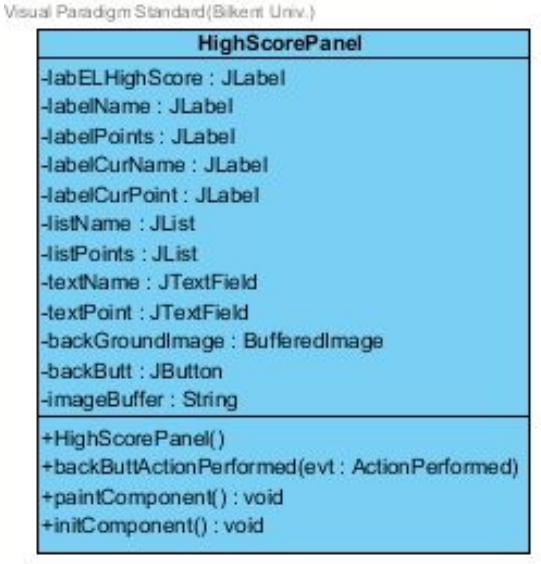
public CreditsPanel() : Constructor for SettingsPanel class to initialize objects.

public void backButtActionPerformed(ActionEvent evt) : Action listener for backButt.

paintComponent () : To draw JPanel

initComponent (): To initialize components.

HighScore Panel Class



Attributes

BufferedImage backGroundImage: The image to display in the background of HighScorePanel.

String imageBuffer : Address of image to be used is defined with this String attribute.

private javax.swing.JButton backButt : In HighScorePanel, a JButton instance is called and it's modified to a button with "to back page" functionality.

private javax.swing.JLabel labelHighScore : This is the panel where String imageBuffer will be printed.

private javax.swing.JLabel labelName : This is the panel where String imageBuffer will be printed.

private javax.swing.JLabel labelPoints : This is the panel where String imageBuffer will be printed.

private javax.swing.JLabel labelCurName : This is the panel where String imageBuffer will be printed.

private javax.swing.JLabel labelCurPoint : This is the panel where String imageBuffer will be printed.

private javax.swing.JList listName : This is the list where scorers names will be printed.

private javax.swing.JList listPoints : This is the list where scorers points will be printed.

private javax.swing.JTextField textName : This is the area which will display the selected scrorer's name.

private javax.swing.JTextField textPoint : This is the area which will display the selected scrorer's points.

Methods

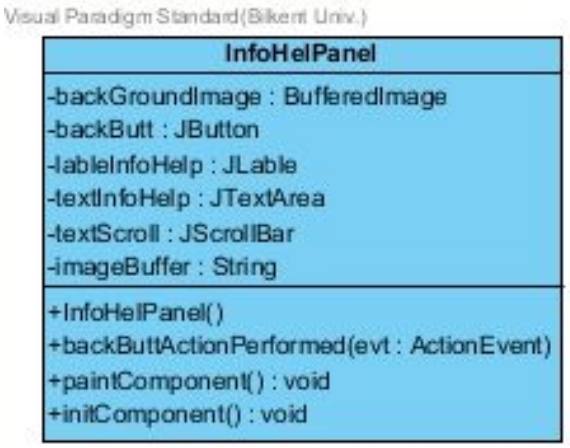
public HighScorePanel() : Constructor for SettingsPanel class to initialize objects.

public void backButtActionPerformed(ActionEvent evt) : Action listener for backButt.

paintComponent () : To draw JPanel

initComponent (): To initialize components.

InfoHelp Panel Class



Attributes

BufferedImage backgroundImage : The actual image to be taken with String attribute imageBuffer.

String imageBuffer : This string holds the address of image to be used as backgroudImage.

private javax.swing.JButton backButt : In HighScorePanel, a JButton instance is called and it's modified to a button with "to back page" functionality.

private javax.swing.JLabel labelName : This is the panel where String imageBuffer will be printed.

private javax.swing.JTextArea textInfoHelp : Text area where info and help instructions will be written.

private javax.swing.JScrollBar textScroll : Scroll bar for JtextArea.

Methods

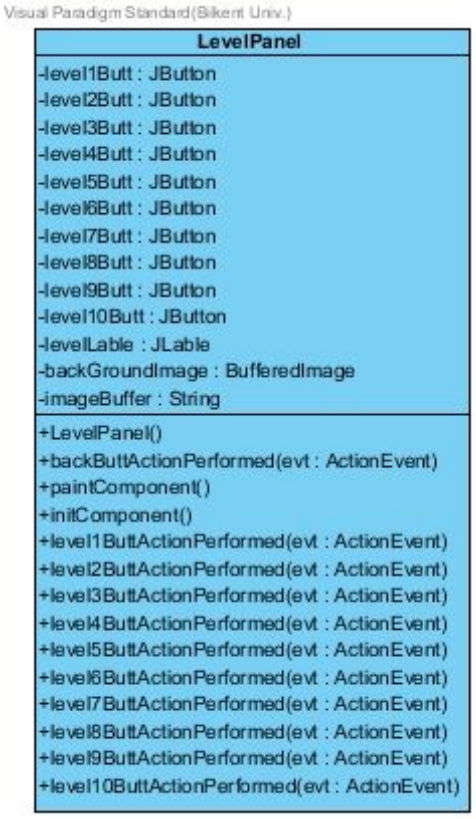
public InfoHelPanel() : Constructor for SettingsPanel class to initialize objects.

public void backButtActionPerformed(ActionEvent evt) : Action listener for backButt.

paintComponent () : To draw JPanel

initComponent (): To initialize components.

Level Panel Class



Attributes

private javax.swing.JButton level(1-10)Butt : JButton instances which calls levels 1 to 10 respectively.

private javax.swing.JLabel levelLabel : This is the panel where String imageBuffer will be printed.

BufferedImage backgroundImage : The actual image to be taken with String attribute imageBuffer.

Methods

public LevelPanel() : Constructor for SettingsPanel class to initialize objects.

public void backButtActionPerformed(ActionEvent evt) : Action listener for backButt.

paintComponent () : To draw JPanel

initComponent (): To initialize components.

public void level(1-10)ButtActionPerformed(ActionEvent evt) : Action listeners for level(1-10)Butts.

9.2.2 Game Logic Design And Subsystems

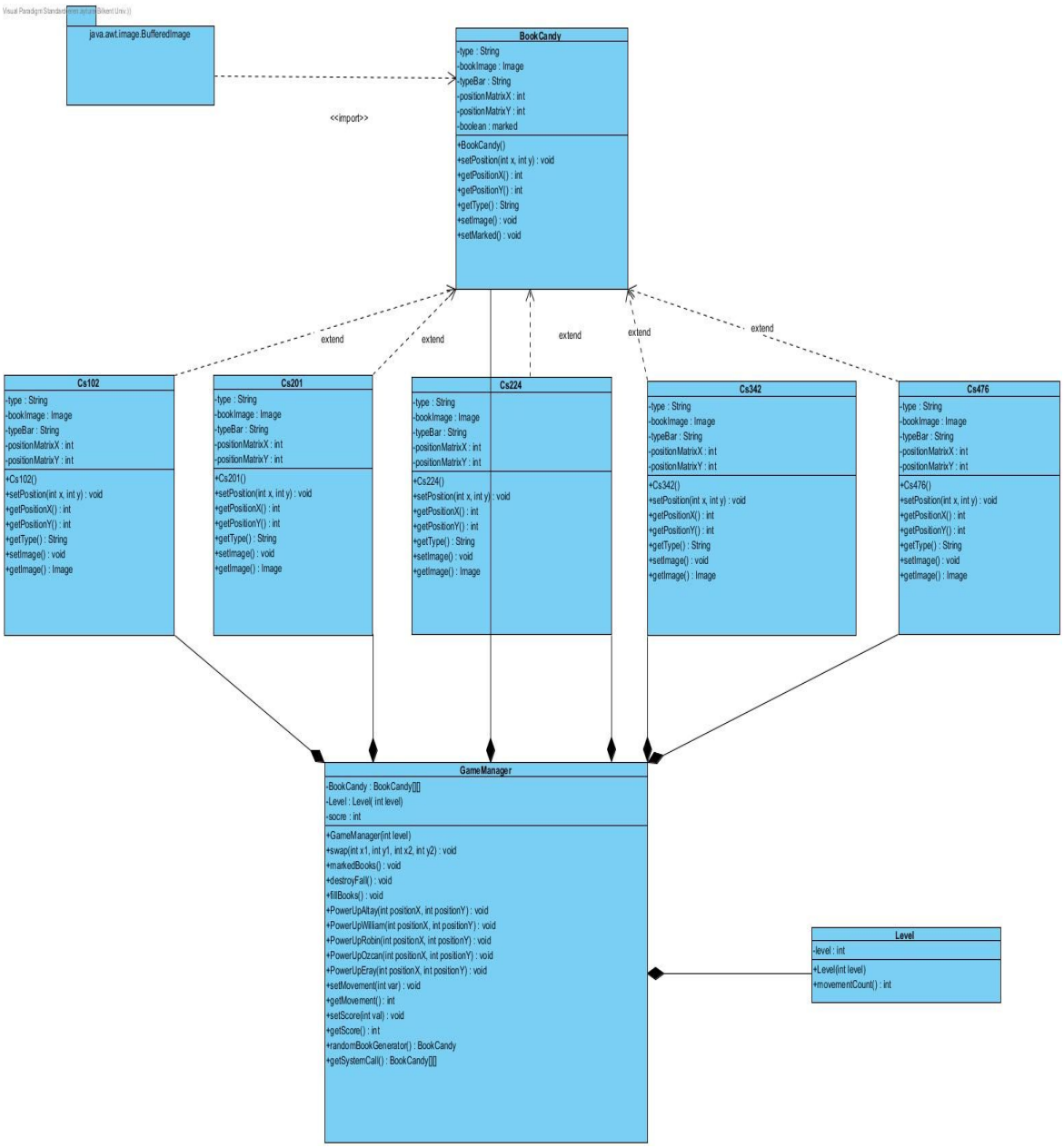
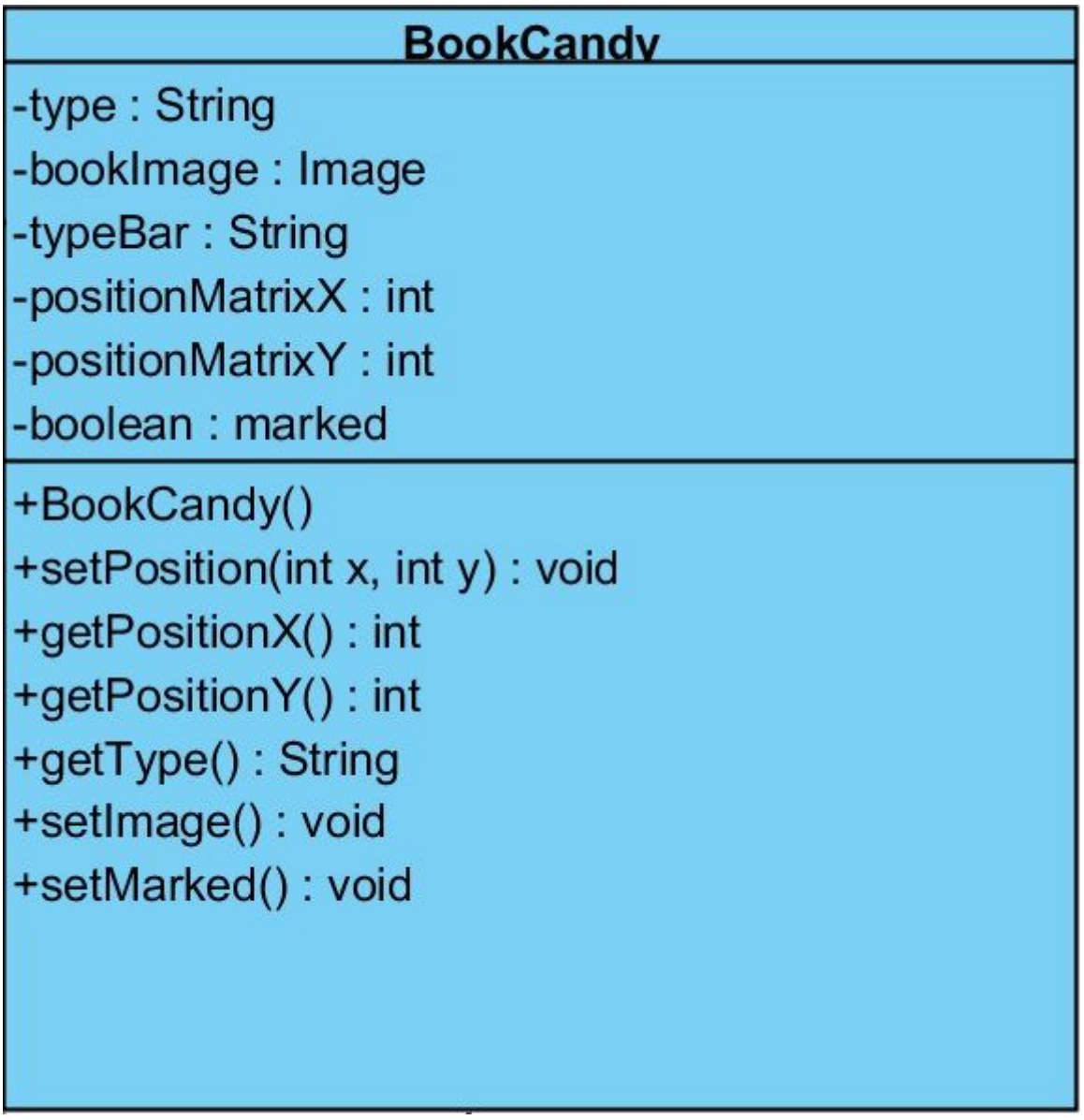


figure 9.2.2

BookCandy Class



Constructor

+BookCandy() : This constructor is the parent of other constructors

Attributes

-private string type: According to this string type the program understands that the order of same types of book is provided or not.

-private Image bookImage: This image shows the type of books to the user.

-private string typeBar: In this game every book has a special type. These types are also splitted vertically or horizontally. This string determines that.

-private int PositionMatrixX: This integer type shows the x coordinate of position.

-private int PositionMatrixY: This integer type shows the y coordinate of position.

-private bool marked: The program understands that the book is special or no.

Methods

+BookCandy(String barType): Vertical, horizontal or normal book.

+public void setPosition(int x, int y): According to swapping or randomly(while starting), this method sets the position of the books.

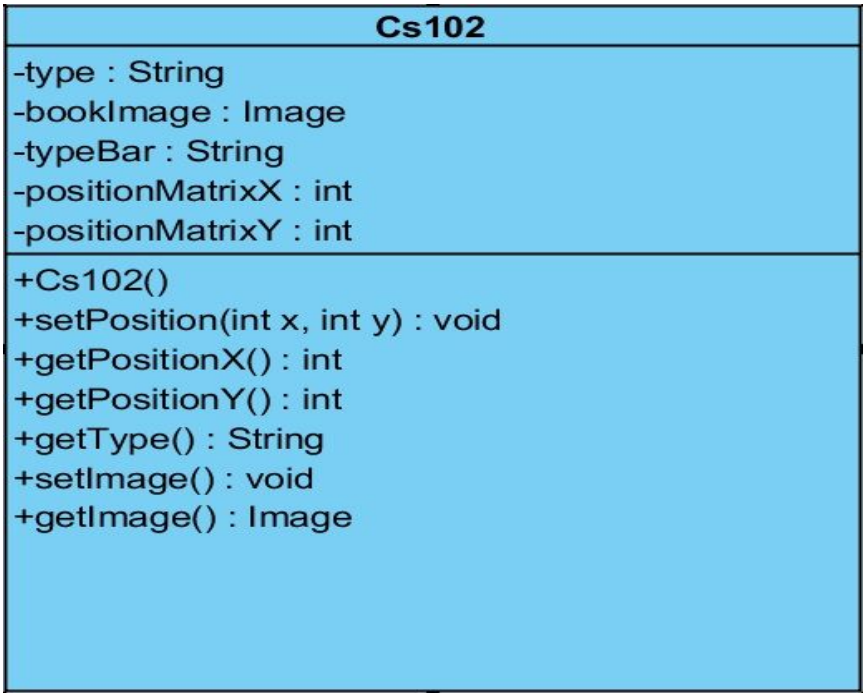
+public int getPositionX(): To disappear or swap the the books, the program should know the x coordinate of the books.

+public int getPositionY(): To disappear or swap the the books, the program should know the y coordinate of the books.

+public string getType(): The program understands the type of books according to the this string.

+public void setImage(): The icons of the books are arranged by using this method.

CS102 Class



Constructor

+Cs102() : The type of the book can be created as a special type in the proportion of this is 5%. If it is special book, the bar type of this book is created as vertical or horizontal randomly.

If it is not special book, it is created normally. On the other hand, the type of this book is created randomly and bar type is arranged “None”.

Attributes

-private string type: The type of this string is is “cs102”.

-private Image bookImage: The icon of this book is “cs102.jpeg”

-private string typeBar: “Vertical”,”Horizontal” or ”None”

-private int PositionMatrixX: The program sets randomly the x coordinate of the book and it is changed by swapping.

-private int PositionMatrixY: The program sets randomly the x coordinate of the book and it is changed by swapping.

-private bool marked: If it is true, it is a special type of cs102 book.

Methods

+public void setPosition(int x, int y): According to swapping or randomly(while starting), this method sets the position of the cs 102 book.

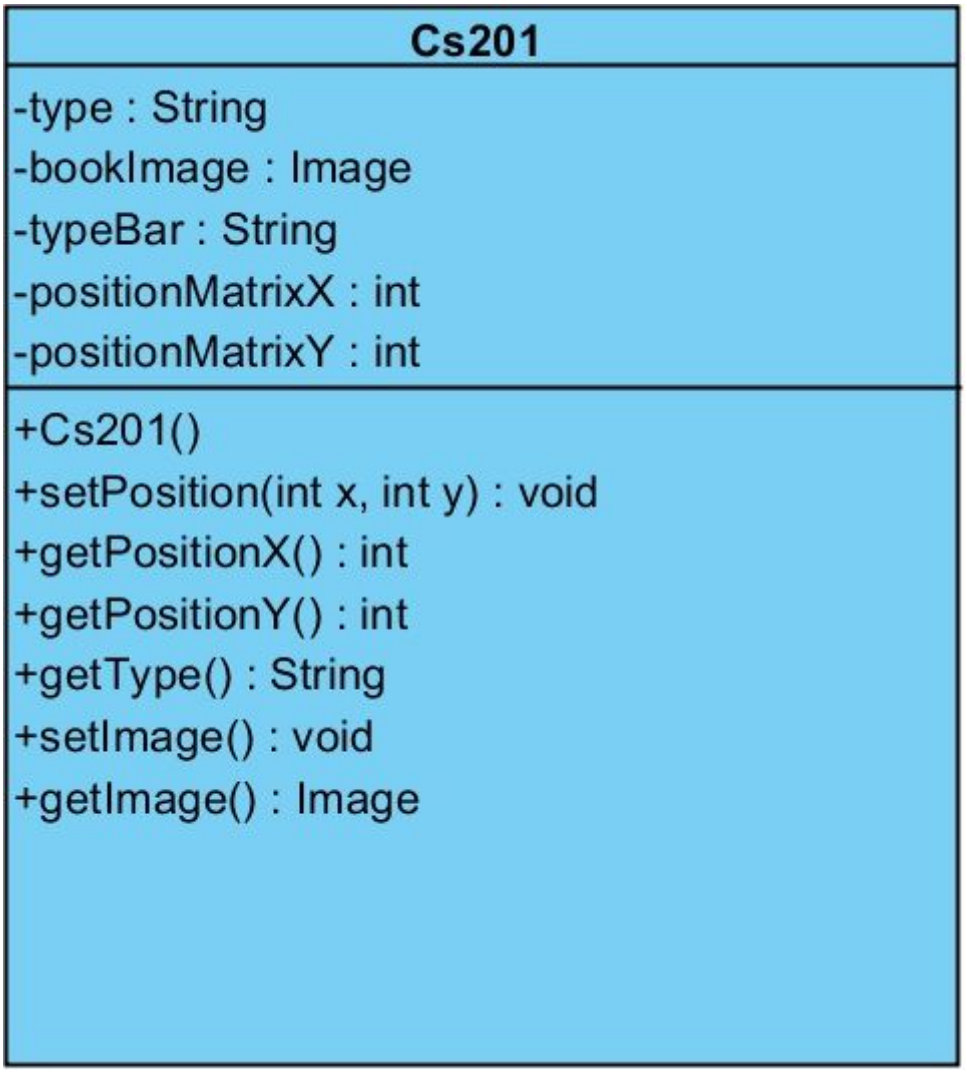
+public int getPositionX(): This method returns the x coordinate.

+public int getPositionY(): This method returns the y coordinate.

+public string getType(): This method returns the type of book as a string.

+public void setImage(): If the gamer uses power up, the type can be special type of the same book. This method changes the normal image with special image of the same book.

CS201 Class



Constructor

+Cs201() : The type of the book can be created as a special type in the proportion of this is 5%. If it is special book, the bar type of this book is created as vertical or horizontal randomly.

If it is not special book, it is created normally. On the other hand, the type of this book is created randomly and bar type is arranged “None”.

Attributes

-private string type: The type of this string is is “cs201”.

-private Image bookImage: The icon of this book is “cs102.jpeg”.

-private string typeBar: “Vertical”,”Horizontal” or ”None”

-private int PositionMatrixX: The program sets randomly the x coordinate of the book and it is changed by swapping.

-private int PositionMatrixY: The program sets randomly the y coordinate of the book and it is changed by swapping.

-private bool marked: If it is true, it is a special type of cs102 book.

Methods

+public void setPosition(int x, int y): According to swapping or randomly(while starting), this method sets the position of the cs 201 book.

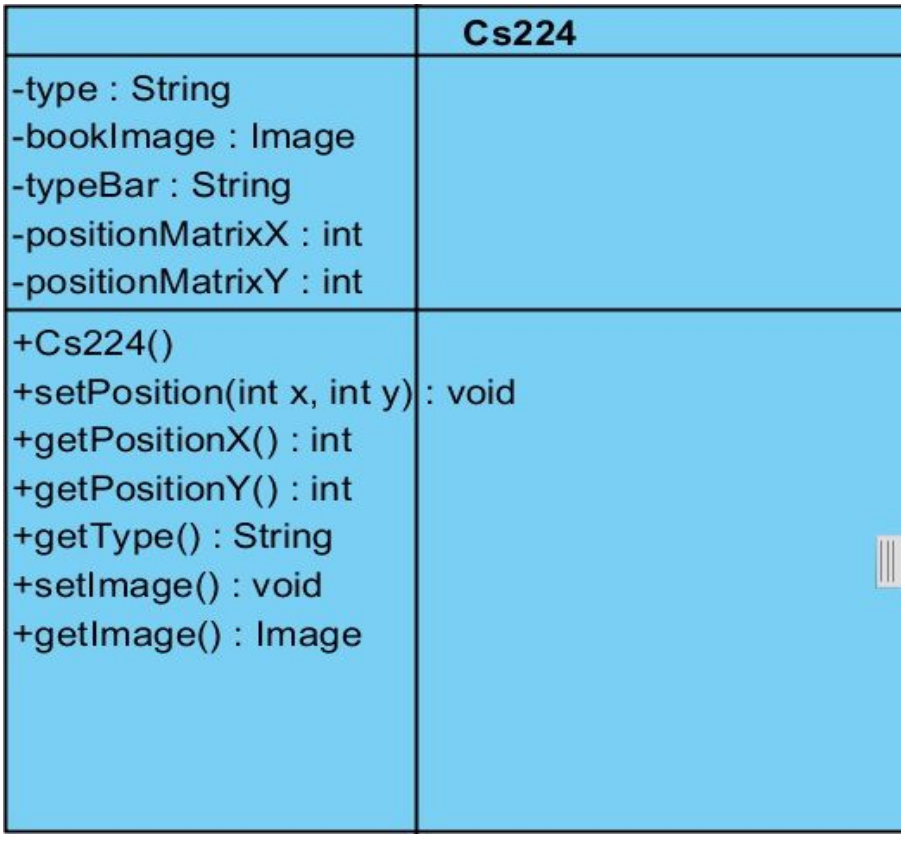
+public int getPositionX(): This method returns the x coordinate.

+public int getPositionY(): This method returns the y coordinate.

+public string getType(): This method returns the type of book as a string.

+public void setImage(): If the gamer uses power up, the type can be special type of the same book. This method changes the normal image with special image of the same book.

CS224 Class



Constructor

+Cs224() : The type of the book can be created as a special type in the proportion of this is 5%. If it is special book, the bar type of this book is created as vertical or horizontal randomly.

If it is not special book, it is created normally. On the other hand, the type of this book is created randomly and bar type is arranged “None”.

Attributes

-private string type: The type of this string is is “cs224”.

-private Image bookImage: The icon of this book is “cs224.jpeg”.

-private string typeBar:

-private int PositionMatrixX: The program sets randomly the x coordinate of the book and it is changed by swapping.Before swapping, the game determines the x coordinate of the book by starting.

-private int PositionMatrixY: The program sets randomly the y coordinate of the book and it is changed by swapping.Before swapping, the game determines the y coordinate of the book by starting.

-private bool marked: If it is true, it is a special type of cs224 book.

Methods

+public void setPosition(int x, int y): According to swapping or randomly(while starting), this method sets the position of the cs 224 book.

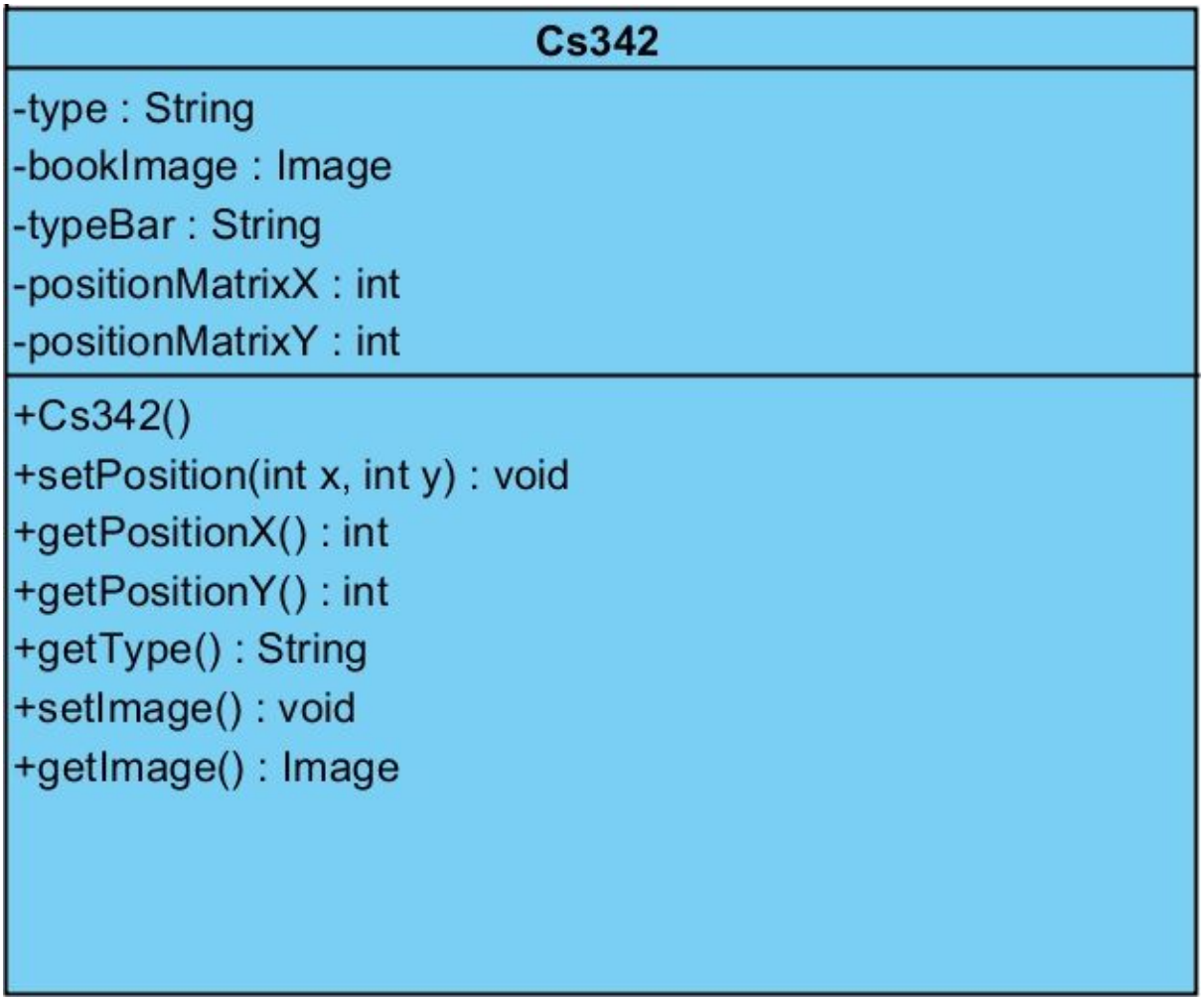
+public int getPositionX(): This method returns the x coordinate.

+public int getPositionY(): This method returns the y coordinate.

+public string getType(): This method returns the type of book as a string.

+public void setImage(): If the gamer uses power up, the type can be special type of the same book. This method changes the normal image with special image of the same book.

CS342 Class



Constructor

+Cs342() : The type of the book can be created as a special type in the proportion of this is 5%. If it is special book, the bar type of this book is created as vertical or horizontal randomly.

If it is not special book, it is created normally. On the other hand, the type of this book is created randomly and bar type is arranged “None”.

Attributes

-private string type: The type of this string is is “cs342”.

-private Image bookImage: The icon of this book is “cs342.jpeg”.

-private string typeBar:“Vertical”,”Horizontal” or ”None”

-private int PositionMatrixX: The program sets randomly the x coordinate of the book and it is changed by swapping.Before swapping, the game determines the x coordinate of the book by starting.

-private int PositionMatrixY: The program sets randomly the y coordinate of the book and it is changed by swapping.Before swapping, the game determines the y coordinate of the book by starting.

-private bool marked: If it is true, it is a special type of cs342 book.

Methods

+public void setPosition(int x, int y): According to swapping or randomly(while starting), this method sets the position of the cs 342 book.

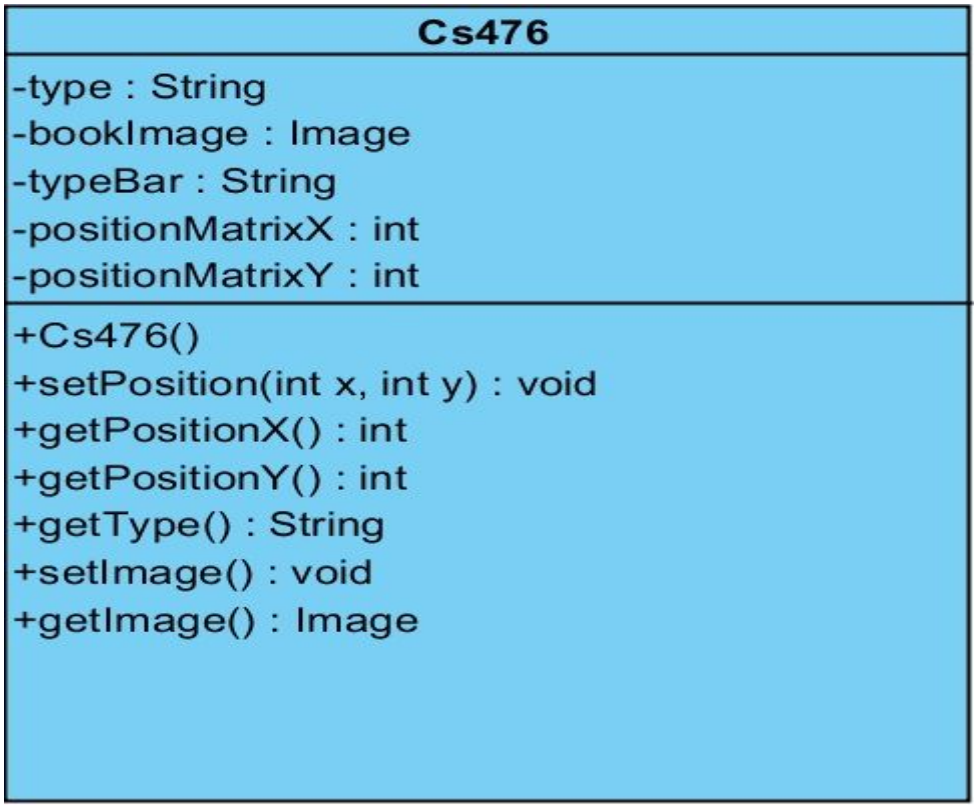
+public int getPositionX(): This method returns the x coordinate.

+public int getPositionY(): This method returns the y coordinate.

+public string getType(): This method returns the type of book as a string.

+public void setImage(): If the gamer uses power up, the type can be special type of the same book. This method changes the normal image with special image of the same book.

CS476 Class



Constructor

+Cs476() : The type of the book can be created as a special type in the proportion of this is 5%. If it is special book, the bar type of this book is created as vertical or horizontal randomly.

If it is not special book, it is created normally. On the other hand, the type of this book is created randomly and bar type is arranged “None”.

Attributes

-private string type: The type of this string is is “cs476”.

-private Image bookImage: The icon of this book is “cs476.jpeg”.

-private string typeBar:

-private int PositionMatrixX: The program sets randomly the x coordinate of the book and it is changed by swapping. Before swapping, the game determines the x coordinate of the book by starting.

-private int PositionMatrixY: The program sets randomly the y coordinate of the book and it is changed by swapping. Before swapping, the game determines the y coordinate of the book by starting.

-private bool marked: If it is true, it is a special type of cs book.

Methods

+public void setPosition(int x, int y): According to swapping or randomly(while starting), this method sets the position of the cs 476 book.

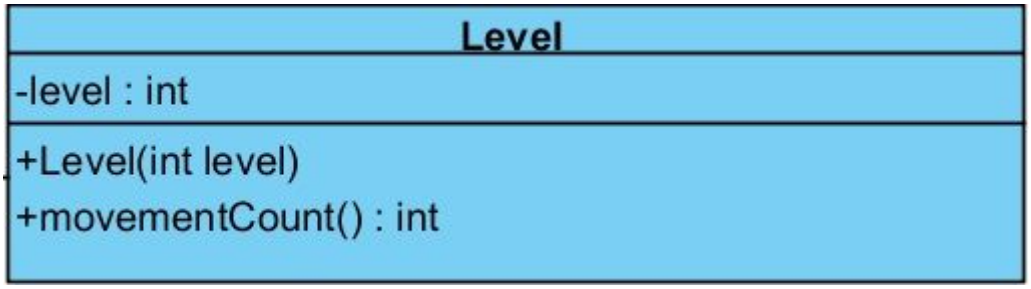
+public int getPositionX(): This method returns the x coordinate.

+public int getPositionY(): This method returns the y coordinate.

+public string getType(): This method returns the type of book as a string.

+public void setImage(): If the gamer uses power up, the type can be special type of the same book. This method changes the normal image with special image of the same book.

Level Class



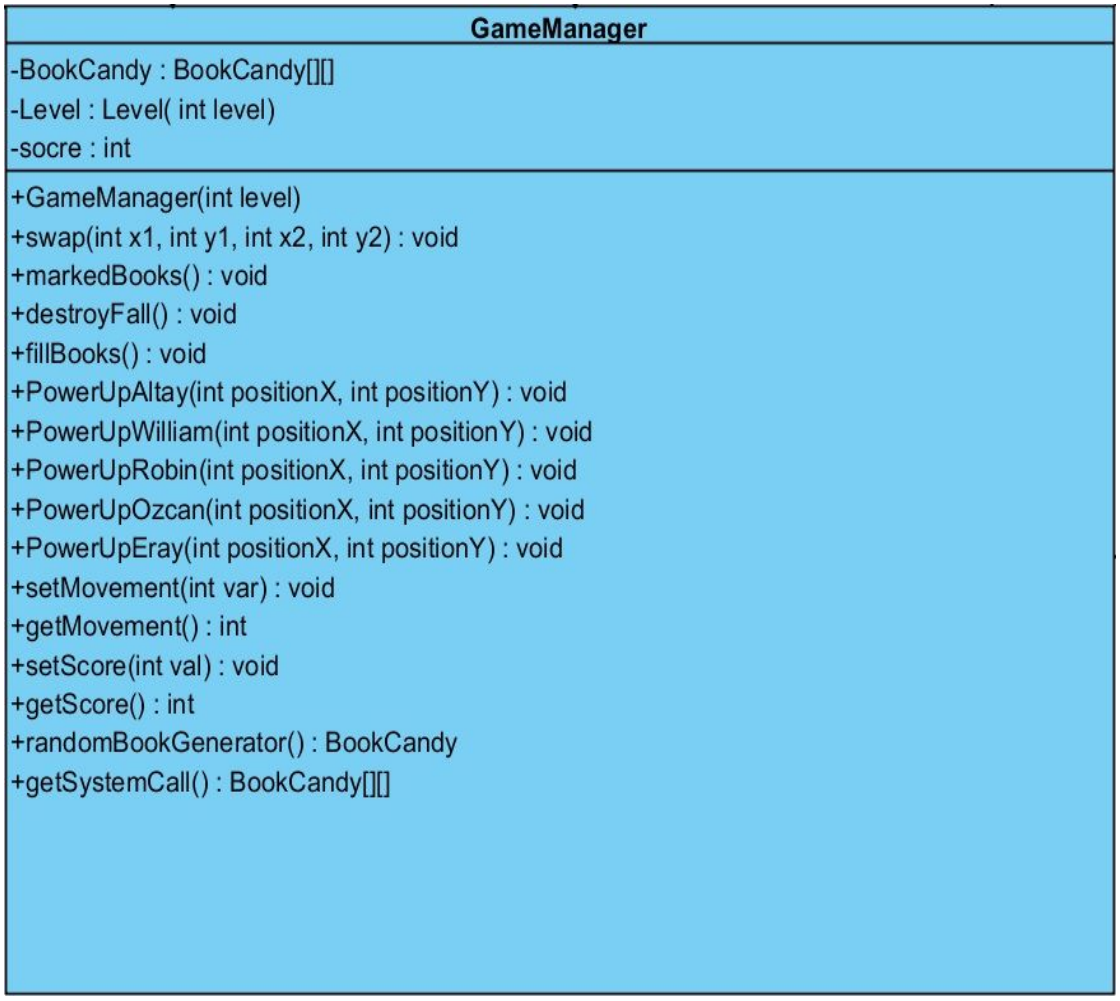
Attributes

-private int level: Determines the level 1 to 10

Methods

+public int movementCount(): Generates movement counts according to level.

GameManager Class



Constructor

BookCandy 2D array is filled by randomBookGenerator() function.

Attributes

-private bookCandy[][]: This array is going to be used for representing the matrix because all visual operations will be checked from this 2D array. This array will hold child classes of BookCandy class. This array will be 10x10 matrix.

-private Level level: This object will generate movement count according to the 1 to 10 level.

-**private int score:** holds the score

Methods

+public void swap(): This method gives permission to swap objects in the margin and 1 next to up,right,left and down(0<=(position+10,position-1,position+1,position-10)<100)).This method calls markedBooks() method, then calls destroyFall() method and fillBooks() Method.

+private void markedBooks(): This method checks if 3 same book either horizontal or vertically and straightly together, then marks them to be destroyed.

+private void destroyFall(): This method checks marked objects in the array then sets null. Null Positions are filled with position-10k(0<k<10) existing objects in the margin of the matrix.

+private void fillBooks(): Null objects are filled with randomBookGenerator() and it stops if loop end or one raw is not filled with anything. This method again calls markedBooks() method, then calls destroyFall() method and fillBooks() method until no object is marked.

+public void PowerUpAltay(int positionX, int positionY): Turns 5 random objects into barType objects.

+public void PowerWilliam(int positionX, int positionY): Inside the 10x10 matrix,8x8 matrix objects can be chosen because this method set these objects and 3x3 matrix is set to null.

+public void PowerUpRobin(int positionX, int positionY): Regenerates a selected position in matrix.

+public void PowerUpOzcan(int positionX, int positionY): Sets one object marked and sets null.After that this method calls, then calls destroyFall() method and fillBooks() Method.

+public void PowerUpEray(int positionX, int positionY): This method can swap any book objects and calls, then calls destroyFall() method and fillBooks() Method.

+public void setMovement(int val): Shifts movement count.

+public int getMovement(): Returns number of movements.

+public BookCandy randomBookGenerator(): Generates books with its type randomly.

+public BookCandy[][] getSystemCall(): Returns 2D BookCandy objects array.

10. References

http://www.wikizero.info/index.php?q=aHR0cHM6Ly9lbi53aWtpcGVkaWEub3JnL3dpa2kvQ2FuZHlfQ3J1c2hfU2FnYQ

http://www.cs.bilkent.edu.tr/~ugur/teaching/cs319/proj/11_2C/analysis.doc

http://www.cs.bilkent.edu.tr/~guvenir/guvenir.jpg

https://media.licdn.com/mpr/mpr/shrinknp_400_400/AAEAAQAAAAAAAAKAAAAAJGRlMWQ1NTBjLTg3YjUtNGI3Mi1hYWI5LTBiOWM5OTQ2NDQ3MA.jpg

https://www.unilica.com/userdata/cff33298abf1a93fe27f238426611fcb7df9857c/m_bc001438aec63666f5b3e94f81cf4f11bba328c1.jpg

https://media-exp2.licdn.com/mpr/mpr/shrinknp_200_200/AAEAAQAAAAAAAAtkAAAAJGUxZWQzNzMwLTkzMjEtNGY2Ni1hNTI2LTZiMzk2YTdhOWM0NQ.jpg

http://bilnews.bilkent.edu.tr/archive/issue_11_30/distinguished_williamsawyer.jpg

https://media.wiley.com/product_data/coverImage300/87/11180878/1118087887.jpg

https://images-na.ssl-images-amazon.com/images/I/61KGTdfkPAL.jpg

https://pictures.abebooks.com/isbn/9780273768418-us-300.jpg

https://secure-ecsd.elsevier.com/covers/80/Tango2/large/9780123944245.jpg

http://codex.cs.yale.edu/avi/os-book/OS9/images/os9c-cover.jpg

https://king.com/tr/game/candycrush

https://en.oxforddictionaries.com/definition/cross-platform

https://www.techopedia.com/definition/5442/java-runtime-environment-jre

https://www.urbandictionary.com/define.php?term=fps