Team: Pro-Grammers

Zahara, Cori, Harris, Meghan

VIEWS

~~LoginView~~

- Displays the login screen and collects username and password input from the user
- Displays login results or error message
- Communicates with the ***userprofileModel***, it communicates the following:
- The ***userprofileModel*** stores the username and password from the ***loginView***

~~MainView~~

- This component starts the game when the user interacts with the play button .
- Only ***openingControlle****r* can communicate with this view. It communicates in the following ways:
- The ***openingControlle****r* can ask the ***loginController*** to run the ***loginView***

~~ProfileSetUpView~~

- Displays the account creation screen and collects new user information
- Displays account creation results or error messages
- Communicates with ***profileSetUpController****:* send new account information entered by the user and receives error messages to display if unsuccessful

HubView

- Displays the user's character, inventory, option to view the shop, and option to play game.

  Serves as the main hub or profiles page where the user can view their stats
- Only the ***hubController*** can communicate with this view in the following ways:
- The ***hubController*** can ask the ***shopController*** to run the ***shopView***
- The ***hubController*** can ask the ***gameController*** to run the ***gameView***
- The ***hubController*** can retrieve data information from the ***userProfileModel*** to display in ***hubView***

ShopView

- This component displays the screen with the shop in order to allow the user to sell their gems and buy new items
- It displays the characters inventory, items being sold, and the amount of “gold” the player has
- Communicates with the ***userprofileModel*** that is storing the information for the inventory
- The ***userprofileModel*** will be updated as the player adjusts their inventory in the ***shopView***
- Communicates with the ***userprofileModel*** to update the gold amount as the user sells / buys items, and displays them in the ***shopView***

~~GameView~~

- Displays the excavation level, player and primary gameplay
- This component receives input from the ***gameController, gamePlayer***, and ***tileManager, mapModel*** and the ***UserProfileModel***

MinigameView

- Displays the excavation bone minigame interface, renders the minigame elements and provides interaction mechanisms for the user while capturing user input during the minigame

- Communicates with ***minigameController*** by sending the user interactions during the minigame which updates for rendering

ScrapbookView

- Displays a skeleton that will change depending on the number of bones the current player profile has.

CharaterSelectionView

- Displays the different character sprite options
- Communicates with the ***CharaterSelectionController*** to allow user to change their character

CONTROLLERS

~~LoginController~~

- Handles the logic for user login and authentication
- Processes user credentials and communicates results back to the view
- Communicates with ***LoginView*** and ***UserprofileModel***

~~MainController~~

- Acts as the main controller for the game and holds all other controllers
- Manages interactions on the opening screen
- Initiates the transition from the opening screen to the next part of the application
- Communicates with ***OpeningView, LoginController,*** **ProfileSetUpController, HubController, ShopController, ProfileSetUpController, GameController and MinigameController.**
- Handles the logic for setting up a new user profile
- Processes new account information and communicates results back to the view
- Communicates with ***ProfileSetUpView*** and ***UserProfileModel***

HubController

- Manages the main hub/profile page interactions
- Provides user data and stats to the view
- Handles navigation from the hub to other parts of the application
- Communicates with ***HubView, UserProfileModel, ShopController, GameController***

ShopController

- Controls the transactions made in the shop, including when the user sells their gems or buys new items, and updating the current inventory accordingly
- Communicates with the ***UserprofileModel*** to update the inventory when the user uses items in the inventory in the shop
- Communicates with the ***UserprofileModel*** to update the gold amount as the user sells / buys items

~~GameController~~

- Controls the main game loop, updates and draw calls for player and **TileManger**
- Communicates visual information to the **GameVeiw** using the **MapModel** , and player info with **UserprofileModel**

~~KeyBoardHandler~~

- Accepts input from keyboard and changes internal state

~~Entity(abstract)~~

- Holds information

~~GamePlayer~~

- Accepts input from the **KeyBoardHandler**
- Communicates with gameController,

~~Tile~~

- Holds tile information

~~TileManager~~

- Accepts input from the user and communicates with **GameController**

~~CollisionChecker~~

- Accepts input from entity and map and communicates with those entities

MinigameController

- Initiates the minigame when certain conditions are met, communicates with MiniGameView

MinigameTwoController

- Initiates the second minigame when the method is called and conditions are met, communicates with MiniGameView
-

MinigameThreeController

- Initiates the third minigame when the method is called and conditions are met, communicates with MiniGameView

Item(interface)

- Holds the different types of items that are able to be stored in the inventory, and therefore accessible throughout the game
- Able to update the items in the inventory as the items are added and removed
- Communicates with the ***UserProfileModel***

LevelGenerator

- Generates int[] to serve as maps and interacts with the MapModel to use these maps

Scorpion

- Extends entity
- Interacts with the Game controller to move and interacts with UserProfileModel to do damage.
- Holds variables associated with the enemy like speed, sprite, and coins lost when player is hit

Snake

- Extends entity
- Interacts with the Game controller to move and interacts with UserProfileModel to do damage.
- Holds variables associated with the enemy like speed, sprite, and coins lost when player is hit

EnemyFactory

- Creates different enemies and interacts with the LevelGenerator to place them in the random level

ScrapbookController

- Interacts with the user profile model to get the player bones, and ScrapbookView to display them

CharaterSelectionController

- Allows user to select or change their character sprite and associated attributes
- Interfaces with the UserProfileModel to update userprofile with their sprite

OTHER CLASSES

Shovel

- Item in the shop that the user can purchase using gems to improve their gameplay
- Implements the ***Item*** interface

Pickaxe

- Item in the shop that the user can purchase using gems to improve their gameplay
- Implements the ***Item*** interface

Jackhammer

- Item in the shop that the user can purchase using gems to improve their gameplay
- Implements the ***Item*** interface

Gems

- Item that users collect throughout the game by breaking blocks
- Are used to sell in return for gold, which can then be used to buy other items
- Implements the ***Item*** interface

MODELS

~~UserprofileModel~~

- Stores player data including username, password, money, and inventory
- Handles saving and loading user data to and from database
- Manages authentication and account creation
- Communicates with ***LoginController, ProfileSetUpController, HubController, ShopController GameController, ShopView, and Item***

~~MapModel~~

- Represents the game map data, including tile layouts
- Provides methods to access and modify map data
- Communicates with ***TileManager, GameController, CollisionChecker, GamePlayer***
-

Sample Tables(primary keys underlined):

User Table

Username	Password	Currentsprite	Coins	Bones
harris	secertpass	2	3500	3
tommy	3@5673	4	300	10

Items

ItemID	Itemname
1	"Shovel"
2	"Pickaxe"

Inventory

ItemID	UserID
1	harris
2	tommy

Sprite

SpriteID	Path	JumpVelocity	RunVelocity	Price
1	"/coco/"	16	8	1000
3	"/sugar/"	20	9	3000

UserSprites

SpirteID	UserID
1	harris
3	tommy