Senior Design Project

The Game – Code Review

High Level Design Report

Burak Erkılıç	**21501035**
Mert Sezer	**21400246**
Okan Şen	**21202377**
Berk Yıldız	**21502040**

Supervisor: Halil Altay Güvenir

Jury Members: Shervin Rahimzadeh Arashloo, Can Alkan

1. **Introduction**

Code review is not a settled culture among software development teams; because it is seen as a bothering task due to some reasons like perceived lack of time to fit code review in, perceived lack of review guidelines, perceived no value in the code review process among team members. Given this problem, we proposed a system named "Code Review Game". As we mentioned in our previous report, the ultimate goal of this system is to adopt the habit of code reviewing to the coders of the future starting from the first step of education and to turn the code review and inspection learning into a more efficient and appealing process and our web based system is designed in a way to achieve gamification and it is providing consistent challenges, perceivably fair playing experiences, lack of stagnation, lack of trivial decisions and at the same time it is providing an opportunity to do code reviewing process in a competitive way. There will be 2 actors that will use our system: teacher(administrator) and student. Also there will be 2 alternative ways in which the student can join the game; he/she can join the game individually or he/she can join the game with a team. For achieving modularity of the application, we have used 3-tier architecture and we have divided the overall system into 3 main modules and for the implementation, we are planning to use web supported languages like AngularJS, CSS, Bootstrap. This report includes detailed information about the low-level design of the Code Review Game. Design tradeoffs, packages and class interfaces of our current system are demonstrated in the current report.

1.1. Object Design Trade-offs

1.1.1 Usability vs Functionality

Our system aims to help students in adopting code review practices and therefore the students should not encounter difficulties during their navigation through the application. If the code review game includes too much functionality, then the use of it by students will automatically be harder. However, we also want students to educate themselves and understand the reason behind their usage of this game like finding an opportunity to review fewer than 400 lines of code at a time, knowing what to look for in code reviews, constructive feedback cycle, learning, and sharing, making code review goals and expectations clear. Therefore keeping the balance between usability and functionality is one of our focuses [1].

1.1.2 Modularity vs Performance

The system should be properly moduled in order to be able to add new functional modules or reuse the existing ones. For this purpose, we are separating the system into 3 tiers. However this separation is causing the independent modules that aren't in direct interaction with each other and this may affect the performance of the system negatively. Also if the speed of code review is slow, the code review process distracts the individuals, it becomes a tedious activity between team members and the team velocity decreases as a result. Therefore we will keep the balance between modularity and performance [2] .

1.1.3 Scalability vs Performance

The system will be able to respond several requests; because several requests will arrive concurrently. For instance, different students can solve code review challenges individually at the same time and this means that results must be available to those students almost at the same time. It is important that serving multiple people at the same time means that maintaining the performance for only one user is challenging.

1.1.4 Compatibility vs Extendibility

Although we are developing a desktop web application, it is possible that in the future, our code review game can be extended to different mediums. For example, a mobile version of the application can be developed and such extensions should be compatible with the current 3-tier architecture of our system.

1.2. Interface Documentation Guidelines

This report follows the standard convention for documentation guidelines. All the class names are named in the standard ‘ClassName’ form. Methods and variables follow the same convention as they are also named as ‘methodName()’ and ‘variableName’. In a complete description of a class, class name comes first, followed by the class attributes, and finally the methods. A detailed layout template is given below:

Class	ClassName
Brief description of class.	
Attributes	
variableName: type	Description
Methods	
methodName()	Brief description of method
methodName (parameter)	Brief description of method

1.3. Engineering Standards

We used UML design principles to describe the classes, subsystems, components, etc; because it is easy to use and it is the most common standard used to represent the structure of the system and its functionalities. [3]

1.4. Definitions, Acronyms, and Abbreviations

- **UML:** Unified Modeling language
- **Bootstrap:** A bootstrap is a framework that initializes the operating system (OS) during startup.
- **AngularJS:** AngularJS is a structural framework for dynamic web apps.

2. Packages

2.1. Controller Package

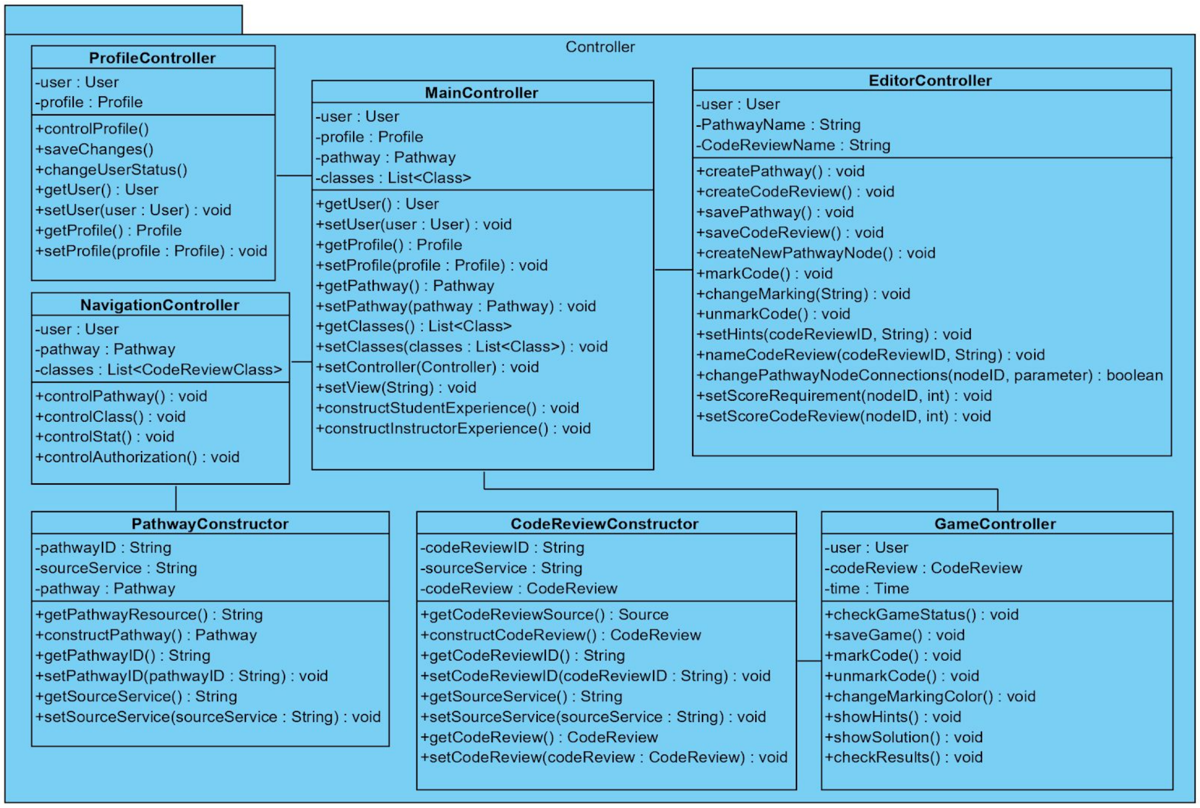

Controller
ProfileController
-user : User
-profile : Profile
+controlProfile()
+saveChanges()
+changeUserStatus()
+getUser() : User
+setUser(user : User) : void
+getProfile() : Profile
+setProfile(profile : Profile) : void
MainController
-user : User
-profile : Profile
-pathway : Pathway
-classes : List<Class>
+getUser() : User
+setUser(user : User) : void
+getProfile() : Profile
+setProfile(profile : Profile) : void
+getPathway() : Pathway
+setPathway(pathway : Pathway) : void
+getClasses() : List<Class>
+setClasses(classes : List<Class>) : void
+setController(Controller) : void
+setView(String) : void
+constructStudentExperience() : void
+constructInstructorExperience() : void
EditorController
-user : User
-PathwayName : String
-CodeReviewName : String
+createPathway() : void
+createCodeReview() : void
+savePathway() : void
+saveCodeReview() : void
+createNewPathwayNode() : void
+markCode() : void
+changeMarking(String) : void
+unmarkCode() : void
+setHints(codeReviewID, String) : void
+nameCodeReview(codeReviewID, String) : void
+changePathwayNodeConnections(nodeID, parameter) : boolean
+setScoreRequirement(nodeID, int) : void
+setScoreCodeReview(nodeID, int) : void
NavigationController
-user : User
-pathway : Pathway
-classes : List<CodeReviewClass>
+controlPathway() : void
+controlClass() : void
+controlStat() : void
+controlAuthorization() : void
PathwayConstructor
-pathwayID : String
-sourceService : String
-pathway : Pathway
+getPathwayResource() : String
+constructPathway() : Pathway
+getPathwayID() : String
+setPathwayID(pathwayID : String) : void
+getSourceService() : String
+setSourceService(sourceService : String) : void
CodeReviewConstructor
-codeReviewID : String
-sourceService : String
-codeReview : CodeReview
+getCodeReviewSource() : Source
+constructCodeReview() : CodeReview
+getCodeReviewID() : String
+setCodeReviewID(codeReviewID : String) : void
+getSourceService() : String
+setSourceService(sourceService : String) : void
+getCodeReview() : CodeReview
+setCodeReview(codeReview : CodeReview) : void
GameController
-user : User
-codeReview : CodeReview
-time : Time
+checkGameStatus() : void
+saveGame() : void
+markCode() : void
+unmarkCode() : void
+changeMarkingColor() : void
+showHints() : void
+showSolution() : void
+checkResults() : void

Figure 1: Controller Package

Controller package includes the classes which provide communication between Presentation and Data tiers. It is the main operator of the core functionality.

- **Profile Controller:** This class holds information of Users and profiles, and additional methods that help control/ edit/ change/ get specific values and users or profiles.
- **Navigation Controller:** This class associates Users with Classes which lets them actively join sessions by giving them a pass.
- **Pathway Constructor:** This class builds the pathway from one end to another end, creating the object of the User’s flow, while also including required methods; such as set, get methods, and constructing a pathway.
- **Main Controller:** This class is where the other classes meet. It holds Users, Profiles, Classes, and the methods required to deal with them.
- CodeReview Constructor: This class holds the CodeReview object’s information. It is directly related to the game controller.
- **Editor Controller:** This class represents the editing part of the game. It includes marking/ unmarking codes, saving, creating new code reviews, setting score requirements, etc..
- **Game Controller:** This class controls the game by keeping track of Users and CodeReviews that are associated in a session.

2.2. Presentation Package

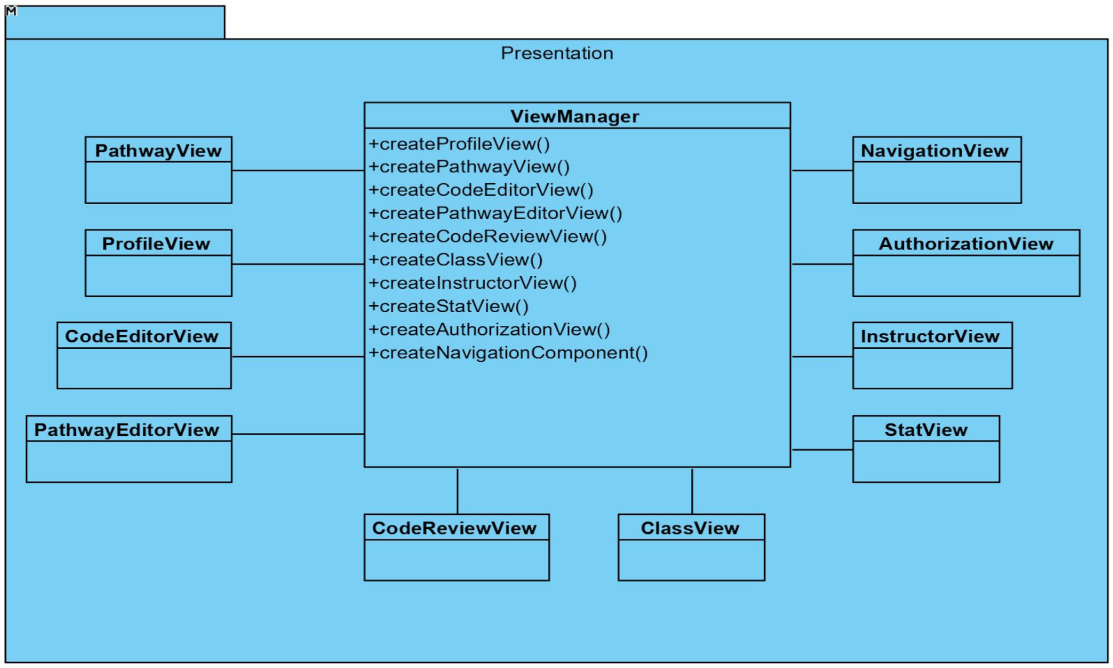
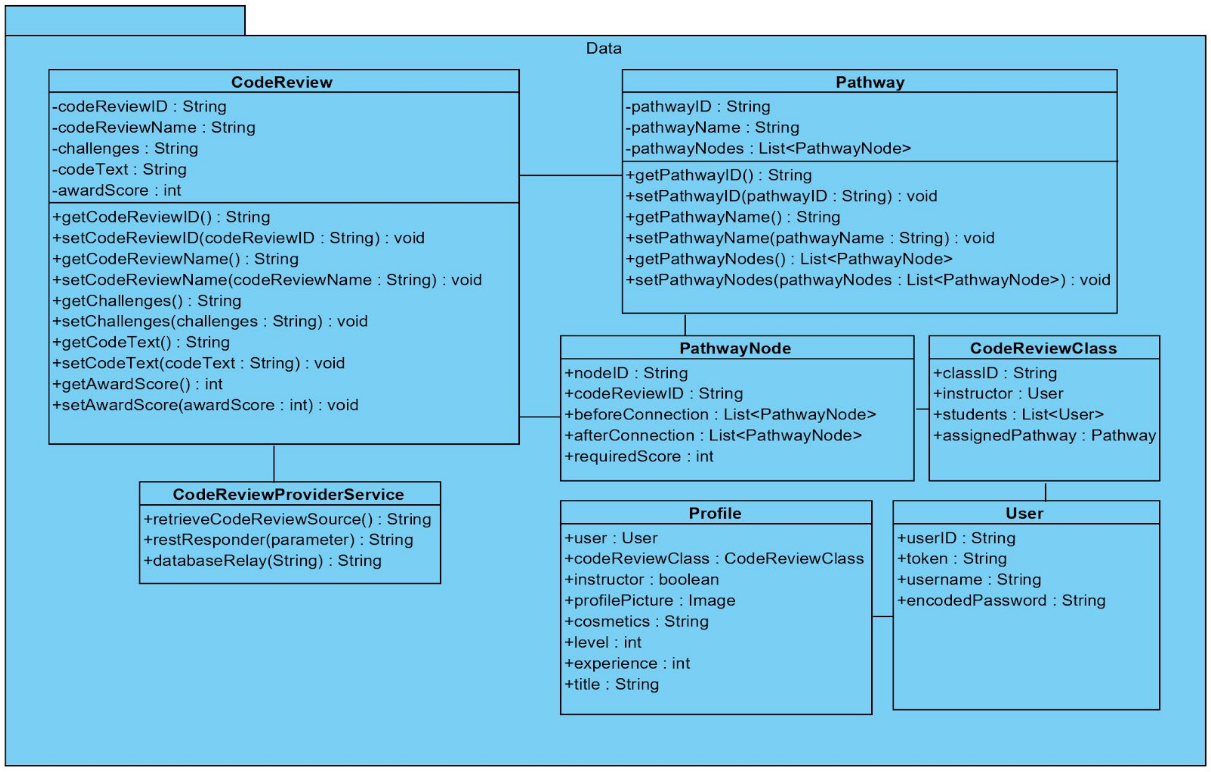


Figure 2: Presentation Package

Presentation package includes the classes which are projected to the users. It is the front end part. It connects the functionality with visibility.

- **ViewManager:** This is the class which connects all views under one roof.
- **Pathway View:** This class shows the pathway screen.
- **Profile View:** This class shows the profile of a user.
- **Code Editor View:** This class shows the code editing screen.
- **Pathway Editor View:** This class shows the pathway editor screen.
- **Code Review View:** This class shows the Code Review game screen.
- **Class View:** This class shows the class which users are gathered in.
- **Stat View:** This class shows the stats of a user.
- **Instructor View:** This class shows the view of an instructor with several more functionalities who has more authorization than a user.
- **Authorization View:** This class shows the authorization screen.
- **Navigation View:** This class shows the navigation in the website.

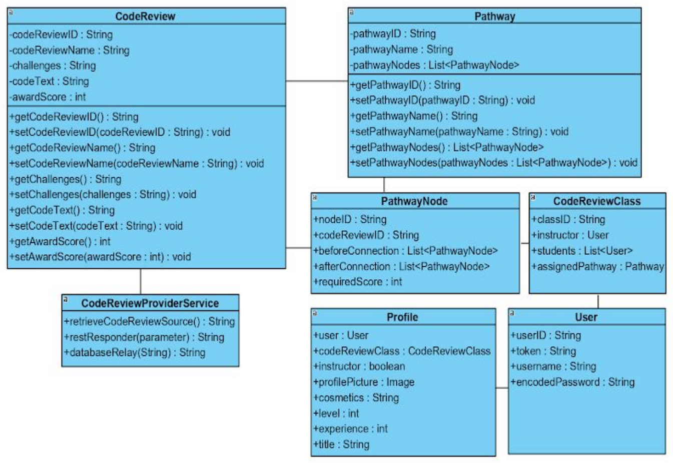
2.2. Data Package

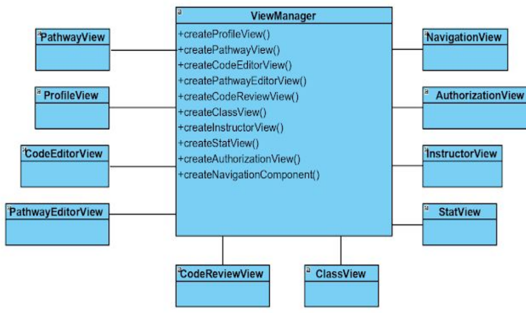

Data
CodeReview
-codeReviewID : String
-codeReviewName : String
-challenges : String
-codeText : String
-awardScore : int
+getCodeReviewID() : String
+setCodeReviewID(codeReviewID : String) : void
+getCodeReviewName() : String
+setCodeReviewName(codeReviewName : String) : void
+getChallenges() : String
+setChallenges(challenges : String) : void
+getCodeText() : String
+setCodeText(codeText : String) : void
+getAwardScore() : int
+setAwardScore(awardScore : int) : void
Pathway
-pathwayID : String
-pathwayName : String
-pathwayNodes : List<PathwayNode>
+getPathwayID() : String
+setPathwayID(pathwayID : String) : void
+getPathwayName() : String
+setPathwayName(pathwayName : String) : void
+getPathwayNodes() : List<PathwayNode>
+setPathwayNodes(pathwayNodes : List<PathwayNode>) : void
PathwayNode
+nodeID : String
+codeReviewID : String
+beforeConnection : List<PathwayNode>
+afterConnection : List<PathwayNode>
+requiredScore : int
CodeReviewClass
+classID : String
+instructor : User
+students : List<User>
+assignedPathway : Pathway
CodeReviewProviderService
+retrieveCodeReviewSource() : String
+restResponder(parameter) : String
+databaseRelay(String) : String
Profile
+user : User
+codeReviewClass : CodeReviewClass
+instructor : boolean
+profilePicture : Image
+cosmetics : String
+level : int
+experience : int
+title : String
User
+userID : String
+token : String
+username : String
+encodedPassword : String

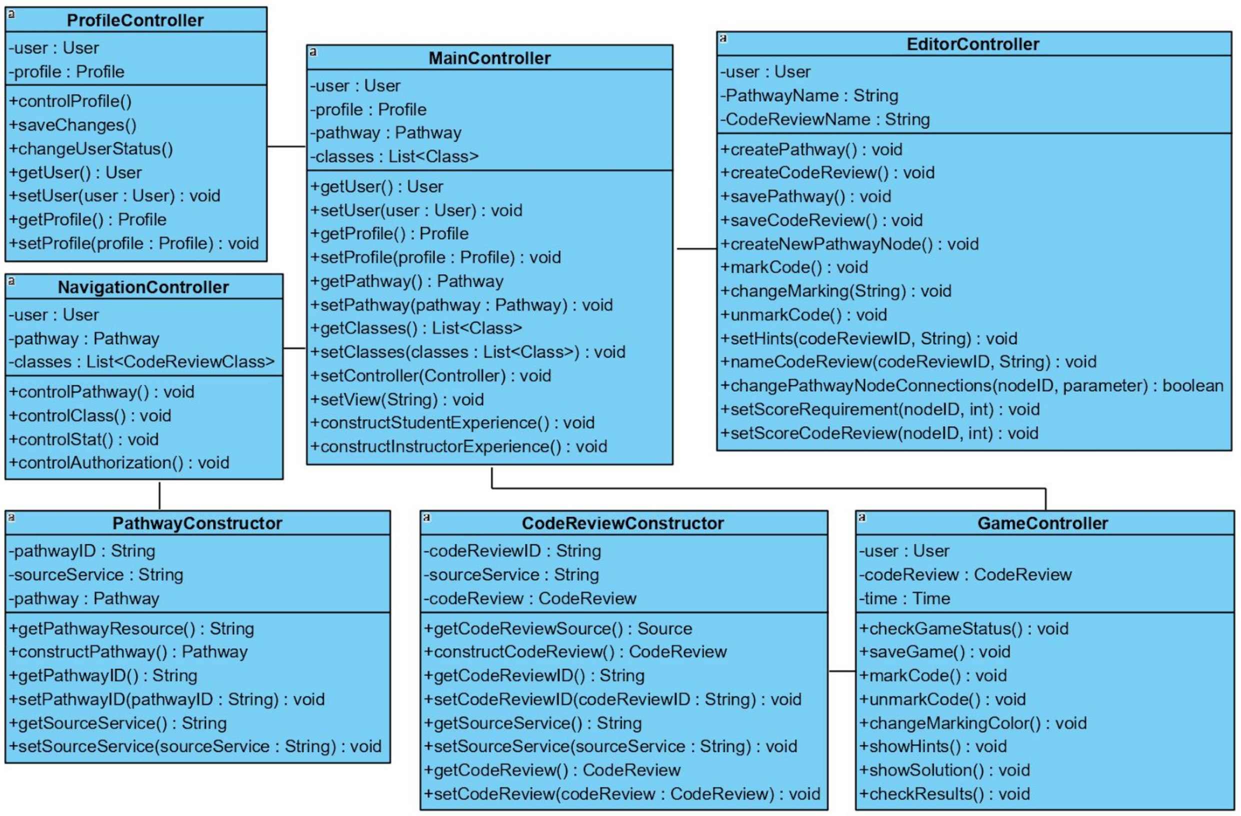
Data package includes the classes which processes and manages data of the application by communicating with the database.

- **CodeReview:** This class holds the object information of CodeReview. Additionally with get and set methods.
- **CodeReviewProviderService**: This class is a subclass of CodeReview and it provides the necessary services to the object.
- **Pathway:** This class holds the object information of Pathway. Additionally with get and set methods.
- **PathwayNode:** This subclass is the node type required for pathway linking.
- **CodeReviewClass:** This class holds the information for the classes which are involved in the code review challenge.
- **Profile:**This class is the object class of profile.
- **User:** This class is related to Profile and it holds the user object data.

3. Class Interfaces


CodeReview
-codeReviewID : String
-codeReviewName : String
-challenges : String
-codeText : String
-awardScore : int
+getCodeReviewID() : String
+setCodeReviewID(codeReviewID : String) : void
+getCodeReviewName() : String
+setCodeReviewName(codeReviewName : String) : void
+getChallenges() : String
+setChallenges(challenges : String) : void
+getCodeText() : String
+setCodeText(codeText : String) : void
+getAwardScore() : int
+setAwardScore(awardScore : int) : void
Pathway
-pathwayID : String
-pathwayName : String
-pathwayNodes : List<PathwayNode>
+getPathwayID() : String
+setPathwayID(pathwayID : String) : void
+getPathwayName() : String
+setPathwayName(pathwayName : String) : void
+getPathwayNodes() : List<PathwayNode>
+setPathwayNodes(pathwayNodes : List<PathwayNode>) : void
PathwayNode
+nodeID : String
+codeReviewID : String
+beforeConnection : List<PathwayNode>
+afterConnection : List<PathwayNode>
+requiredScore : int
CodeReviewClass
+classID : String
+instructor : User
+students : List<User>
+assignedPathway : Pathway
CodeReviewProviderService
+retrieveCodeReviewSource() : String
+restResponder(parameter) : String
+databaseRelay(String) : String
Profile
+user : User
+codeReviewClass : CodeReviewClass
+instructor : boolean
+profilePicture : Image
+cosmetics : String
+level : int
+experience : int
+title : String
User
+userID : String
+token : String
+username : String
+encodedPassword : String




ProfileController
-user : User
-profile : Profile
+controlProfile()
+saveChanges()
+changeUserStatus()
+getUser() : User
+setUser(user : User) : void
+getProfile() : Profile
+setProfile(profile : Profile) : void
MainController
-user : User
-profile : Profile
-pathway : Pathway
-classes : List<Class>
+getUser() : User
+setUser(user : User) : void
+getProfile() : Profile
+setProfile(profile : Profile) : void
+getPathway() : Pathway
+setPathway(pathway : Pathway) : void
+getClasses() : List<Class>
+setClasses(classes : List<Class>) : void
+setController(Controller) : void
+setView(String) : void
+constructStudentExperience() : void
+constructInstructorExperience() : void
EditorController
-user : User
-PathwayName : String
-CodeReviewName : String
+createPathway() : void
+createCodeReview() : void
+savePathway() : void
+saveCodeReview() : void
+createNewPathwayNode() : void
+markCode() : void
+changeMarking(String) : void
+unmarkCode() : void
+setHints(codeReviewID, String) : void
+nameCodeReview(codeReviewID, String) : void
+changePathwayNodeConnections(nodeID, parameter) : boolean
+setScoreRequirement(nodeID, int) : void
+setScoreCodeReview(nodeID, int) : void
NavigationController
-user : User
-pathway : Pathway
-classes : List<CodeReviewClass>
+controlPathway() : void
+controlClass() : void
+controlStat() : void
+controlAuthorization() : void
PathwayConstructor
-pathwayID : String
-sourceService : String
-pathway : Pathway
+getPathwayResource() : String
+constructPathway() : Pathway
+getPathwayID() : String
+setPathwayID(pathwayID : String) : void
+getSourceService() : String
+setSourceService(sourceService : String) : void
CodeReviewConstructor
-codeReviewID : String
-sourceService : String
-codeReview : CodeReview
+getCodeReviewSource() : Source
+constructCodeReview() : CodeReview
+getCodeReviewID() : String
+setCodeReviewID(codeReviewID : String) : void
+getSourceService() : String
+setSourceService(sourceService : String) : void
+getCodeReview() : CodeReview
+setCodeReview(codeReview : CodeReview) : void
GameController
-user : User
-codeReview : CodeReview
-time : Time
+checkGameStatus() : void
+saveGame() : void
+markCode() : void
+unmarkCode() : void
+changeMarkingColor() : void
+showHints() : void
+showSolution() : void
+checkResults() : void

3.1 Presentation Tier Interfaces

Class	View Manager
This class operates and controls the appearance of components and screens.	
Methods	
createProfileView()	Creates view of ‘Create Profile’ view which user creates profile
createPathwayView	Creates view of ‘Create Pathway’ view which instructor creates pathway
createCodeEditorView()	Creates view of the code editor
createPathwayEditorView()	Creates view of the pathway editor
createCodeReviewView()	Creates view of ‘Code Review’ screen
createClassView()	Creates view of a class
createInstructorView()	Creates view of ‘Instructor’ screen
createStatView()	Creates view of ‘Statistics’ screen
createAuthorizationView()	Creates view of login component.
createNavigationComponent()	Creates view of navigation component.

Class	PathwayView
This class includes the attributes of the ‘Pathway’ screen.	

Class	ProfileView
This class includes the attributes of the ‘Profile’ screen.	

Class	CodeEditorView
This class includes the attributes of the ‘Code Editor’ screen.	

Class	PathwayEditorView
This class includes the attributes of the ‘Pathway Editor’ screen.	

Class	CodeReviewView
This class includes the attributes of the ‘Code Review’ screen.	

Class	ClassView
This class includes the attributes of the ‘Class’ screen.	

Class	StatView
This class includes the attributes of the ‘Statistics’ screen.	

Class	InstructorView
This class includes the attributes of the ‘Instructor’ screen.	

Class	AuthorizationView
This class includes the attributes of the ‘Login’ screen.	

Class	NavigationView
This class includes the attributes of the ‘Menu’ screen.	

3.2 Data Tier Interfaces

Class	CodeReview
This class holds the code review challenges.	
Attributes	
codeReviewID: String	Identical id of the code review that is string variable
codeReviewName: String	Identical string code review name
challenges: String	String name of the challenge
codeText: String	String texture of code review
awardScore: int	Must score for players to get award
Methods	
getCodeReviewID()	Gets the identical ID of the code review
setCodeReviewID (String codeReviewID)	Sets an identical ID to the code review
getCodeReviewName	Gets the identical name of the code review
setCodeReviewName (String codeReviewName)	Sets an identical name to the code review
getChallenges()	Returns string names(code review name) of the challenges
setChallenges (String challenges)	Sets string names(code review name) of the challenges
getCodeText()	Returns string texture of code review
setCodeText (String codeText)	Sets source code of the code review challenge.
getAwardScore()	Returns an integer variable that is the must score of the players to get award
setAwardScore (int awardScore)	Sets an integer to the variable that is the must score of the players to get award

Class	Profile
This class states the attributes for a profile.	
Attributes	
user: User	Account that using the system through posting challenges or through solving the challenges
codeReviewClass: CodeReviewClass	Class which user belongs to
instructor: boolean	Variable to check whether the user is instructor or not
profilePicture: Image	Profile picture of the user
cosmetics: String	Cosmetics that user own
level: int	Indicate the level of user
experience: int	Indicate the experience point that user gained
title: String	Title of the user

Class	CodeReviewProviderService
This class provides connection and management with database.	
Methods	
retrieveCodeReviewSource()	Retrieves and returns the string code review source
restResponder()	Handles REST responses
databaseRelay()	Provides connection between database and application

Class	PathwayNode
This class holds attributes of a single code review challenge in a pathway.	
Attributes	
nodeID: String	Unique ID of the pathway node
codereviewID: String	Unique code review id of the pathway
beforeConnection: List<PathwayNode>	List of pathway nodes before the connection is established
afterConnection: List<PathwayNode>	List of pathway nodes after the connection is established
requiredScore: int	Required score that is required for completing the pathway

Class	CodeReviewClass
This class states attributes for code review class.	
Attributes	
classID: String	ID of the class in which the code review game will be played
Instructor: User	Administrator that is responsible for posting the challenges
Students: List<User>	List of students that are able to solve the code review challenges
assignedPathway: Pathway	Pathway that is assigned by the administrator of the game

Class	Pathway
This class holds and manages data for pathways.	
Attributes	
pathwayID: String	Unique ID of the pathway that is string
pathwayName: String	Unique name of the pathway that is string
pathwayNodes: List<PathwayNode>	List of pathway nodes
Methods	
getPathwayID()	Gets and returns the unique ID of the pathway
setPathwayID (String PathwayID)	Sets a string variable for the pathwayID
getPathwayName()	Gets and returns the name of the pathway
setPathwayName (String PathwayName)	Sets a string variable for the name of the pathway
getPathwayNodes()	Returns all nodes on the PathwayNode list
setPathwayNodes (List <PathwayNode> pathwayNodes)	Receives the pathway node list in its parameter to form the list of pathway nodes

Class	Profile
This class states attributes for profiles.	
Attributes	
userID: String	Unique ID of the user that is string
Token: String	Access token that contains the security credentials for a login session
username: String	Unique name of the user that is string
encodedPassword: String	Encoded password of the user

3.3 Controller Tier Interfaces

Class	MainController
This class is the main operator of the application by controlling users, profiles, pathways and classes.	
Attributes	
user: User	Account that using the system through posting challenges or through solving the challenges
profile: Profile	Profile of a particular user
pathway: Pathway	A pathway which consists of several code review challenges
classes: List<Class>	List of classes in the application
Methods	
getUser()	Returns user
setUser (user)	Sets user
getProfile()	Returns profile
setProfile (profile)	Sets profile
getPathway()	Return pathway
setPathway (pathway)	Sets pathway
getClasses()	Return classes
setClasses (classes)	Sets classes
setController (controller)	Sets controller
setView (String)	Sets view of the applicationBu
constructStudentExperience	Builds student experience by considering user's action in the code review challenges
constructInstructorExperience	Builds instructor experience by considering instructor's action in the application

Class	EditorController
This class controls and manages the operations on pathways and code editor.	
Attributes	
user: User	Account that using the system through posting challenges or through solving the challenges
PathwayName: String	Name of the pathway which user is in
CodeReviewName: String	Name of the code review challenge which user solves
Methods	
createPathway()	Creates pathway
createCodeReview()	Creates code review challenge
savePathway()	Saves pathway in to database
saveCodeReview()	Saves code review challenge in to database
createNewPathwayNode()	Creates new code review challenge for a pathway
markCode()	Marks a code snippet on the code editor
changeMarking (String)	Changes the marking on a code snippet
unmarkCode	Unmarks code snippet
setHints (String codeReviewID)	Set hints for a code review challenge
nameCodeReview (String codeReviewID)	Set name for the code review challenge
changePathwayNodeConnections (nodeID, parameter)	Rearrange the path of a pathway
setScoreRequirement (nodeID)	Set score requirements of code review challenge for user to achieve
setScoreCodeReview (nodeID)	Set maximum score for a code review challenge

Class	GameController
This class operates the gamification aspects of the application.	
Attributes	
user: User	Account that using the system through posting challenges or through solving the challenges
codeReview: CodeReview	Code review challenge
time: Time	Duration of the game
checkGameStatus()	Check if the game continues or finishes
saveGame()	Saves the progress
markCode()	Marks code snippet on the code editor
unmarkCode()	Unmarks code snippet on the code editor
changeMarkingColor()	Change marking color on the editor
showHints()	Shows hints of the code review challenge
showSolution()	Shows solution of the code review challenge
checkResults()	Calculates score of the student

Class	ProfileController
This class manages the profile operations.	
Attributes	
user: User	Account that using the system through posting challenges or through solving the challenges
profile: Profile	Profile of a particular user.
Methods	
controlProfile()	Checks the profile
saveChanges()	Save the changes on profile
changeUserStatus()	Changes the user status if there is a need
setPathwayID (pathwayID)	Sets unique ID to a pathway
getUser()	Returns the user
setUser (user)	Sets a user
getProfile()	Returns the profile of the user
setProfile(profile)	Sets a profile to the user

Class	NavigationController
This class provides the navigation in the application	
Attributes	
user: User	Account that using the system through posting challenges or through solving the challenges
pathway: Pathway	Profile of a particular user
classes: List<CodeReviewClass>	List of classes which are assigned for code review challenge(s)
Methods	
controlPathway()	Navigation for pathways
controlClass()	Navigation classes
controlStat()	Navigation for statistics of the accounts
controlAuthorization()	Navigation for login operations

Class	PathwayConstructor
This class builds pathways of the application	
Attributes	
pathwayID: String	Unique ID for a pathway
sourceService: String	Source code service
Pathway: Pathway	Pathway
getPathwayResource()	Returns pathway resource
constructPathway()	Builds pathway
getPathwayID()	Return ID of pathway
setPathwayID (pathwayID)	Sets ID for pathway
getSourceService	Returns source code
setSourceService (String sourceService)	Sets source code

Class	CodeReviewConstructor
This class builds code review challenges.	
Attributes	
codeReviewID: String	Unique ID for code review challenge
sourceService: String	Source code service for code review challenge
codeReview: CodeReview	Code review challenge object
getCodeReviewSource()	Returns source code of code review challenge
constructCodeReview()	Builds code review challenge
getCodeReviewID()	Returns code review challenge ID
setCodeReviewID (String codeReviewID)	Sets code review challenge ID
getSourceService	Returns source code service
setSourceService (String sourceService)	Sets source code service
getCodeReview()	Returns code review challenge
setCodeReview (CodeReview codeReview)	Sets code review challenge

4. Glossary

- **Code Review Challenge:** The main point of the game. Gaining scores by finding
- defects in a code snippet in a limited time.
- **Pathway:** Group of code review challenges which are ordered in difficulty levels.
- **Cosmetics:** Achievements that user gains according to his/her performance in code review
- challenge.

5. References

[1] “9 Best Practices for Code Review”, December 4, 2018. [Online]. Available: https://www.perforce.com/blog/qac/9-best-practices-for-code-review

[2] “11 proven practices for more effective, efficient peer code review”, January 25, 2011. [Online]. Available: https://www.ibm.com/developerworks/rational/library/11-proven-practices-for-peer-review/

[3] IBM, "UML - Basics," June 2003. [Online]. Available: http://www.ibm.com/developerworks/rational/library/769.html