# 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

## 1.1. Purpose of the System

Code review and inspection are important parts of the software development; because they improve code quality and they make codebase more stable and therefore they add value to the software system [1]. They also enhance the knowledge of the developers involved and give developers an opportunity to acquire skills and boost the performance of their code and code practices being well understood within team members is important for companies that want to grow and accelerate delivery.[2] If code review is done badly, it can have a negative impact on team performance and atmosphere 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 and because practical approach is missing toward the code review, code review is seen as a bothering task among most of the software developers.[3] In addition to that, code review is not seen as a regular phase in software development lifecycle and therefore is not a settled culture among less experienced developers. The aim of our code review game is changing the non-practical approaches and adopting the habit of code reviewing to the coders of the future starting from the first step of education and turning the code review and inspection learning into a more efficient and appealing process. To achieve this, we will build a web based game and with that game, code inspection and revision processes will be simulated in both a group level and on an individual level. To achieve the gamification, the application will consist of elements like consistent challenges, perceivably fair playing experiences, lack of stagnation, lack of trivial decisions. To promote teamwork and to increase the sense of togetherness, our game will provide reviewing the code scripts in a competitive way. To achieve the web based environment goal, the application will be developed using web supported languages and will be implemented in servers available online.

## 1.2. Design Goals

**Readability:** Readability in software refers to the understandability and roles of specific functions, methods, or classes that must be understandable to satisfy readability. Because good descriptive names make code easier to understand, the game will give feedback on names that are abbreviated or difficult to understand.

**Performance:** Code reviews are tedious for most of the developers and take time to do it properly and because the level of performance experienced by developers affects their overall satisfaction, it is important for the system to respond in an acceptable amount of time(less than a second) for any sort of request [4].

**Usability:** The system is about practices of code review that is a sensitive issue for the young developers. In order to encourage learning, the system will have a user friendly interface and a user will be able to use all facilities in his/her first meet with the program by our basic navigational path.

**Security:** The system needs to be capable of protecting personal data of individual users and since there is a login function, all passwords and email addresses are needed to be well protected.

**Portability:** The Game will be a web program and going to work in all browsers with any operating system on desktop.

**Availability:** There is no need of extra installation to run program. Program is accessible from all types of internet browsers.

**Scalability:** System will be able to respond to several requests because especially in classwork, several requests will arrive concurrently.

## 1.3. Definitions, acronyms, and abbreviations

Three-tier architecture - A **three-tier architecture** is a client-server **architecture** in which the functional process logic, data access, computer data storage, and user interface are developed and maintained as independent modules on separate platforms.

HTML5 - **HTML5** is a W3C specification that defines the fifth major revision of the Hypertext Markup Language (HTML).

CSS - Cascading Style Sheets.

Bootstrap - A **bootstrap** is a framework that initializes the operating system (OS) during startup.

AngularJS - **AngularJS** is a structural framework for dynamic web apps.

Spring - The **Spring** Framework is an application framework and inversion of control container for the **Java** platform.

Hibernate - The **Hibernate** is an application framework and inversion of control container for the Java platform.

REST API - A RESTful API is an application program interface (API) that uses HTTP requests to GET, PUT, POST and DELETE data

Django - **Django** is an advanced Web framework written in Python that makes use of the model view controller (MVC) architectural pattern.

HTTPS protocol - HyperText Transfer Protocol Secure

SHA256 - **SHA256** stands for Secure Hash Algorithm 256-bit and it's used for cryptographic security.

Cookies - A small text file (up to 4KB) created by a website that is stored in the user's computer either temporarily for that session only or permanently on the hard disk (persistent cookie).

Sessions - In **computer** science and networking in particular, a **session** is a temporary and interactive information interchange between two or more communicating devices, or between a **computer** and user

Firewall - Firewalls are security systems designed to prevent unauthorised access to or from your computer or private network.

## 1.4. Overview

Code review game is a desktop web program that includes the actors' teacher who posts the challenges and the student who solves the challenges and aim of the game is to make code review a natural member of the education process by

using gamification methods. In the game, possible use case scenarios for a teacher are to prepare a review bank, prepare pathway and manage classes and possible use case scenarios for a student are login, register, preview profile, preview pathway, solve code review problem, preview code review solution.
Student can enter challenges individually and can pinpoint defects, pick/unpick defects. Teacher can upload the code script to the system by indicating the name of the programming language and specifying the defects on the code and can create a pathway; in where students won't be able to skip a level without solving the current one. There are 2 types of game modes that are challenge(single player version) in where student does an individual attempt in the game and the results are immediately available and team(multiplayer version) in where student does start a team or join a team that already exists and he/she waits for everyone else to finish for the results to be available. For the implementation, the technologies
will be selected based on the reach of their features and the needs of the application, namely web support and compatibility with most devices as possible.

# 2. Current Software Architecture

Currently, there are not any similar applications on the market. Still, there are several applications offer serious game experience on software development. Those applications can't be seen as a competitor but offer us a similar technological perspective. Our application is based on code knowledge and enriching this knowledge with serious game experience.

## 2.1. HackerRank

This application offers a service in which developers can rank themselves. It is based on code knowledge and problem-solving skills. These two knowledge are directly affecting our product also. Therefore, a basic understanding of competition in software development provided us as a model for our product.

### 2.2. Codingbat

This website is created for problem-solving for developers. It has very simple game mechanics on it which is good enough for us to consider Codingbat as a similar technology. Even though it is developed for enhancing problem-solving still it has a pathway system that we are inspired on.

### 2.3. Code Combat

This is another good example of the fusion between games and software development. Codecombat is a good application for teaching people how to code. Its gamification on code writing inspired us on the development of our own serious game. It can be seen as a beginner for coding version of our product.

## 3. Proposed Software Architecture

### 3.1. Overview

The system is divided into smaller components and according to the “Three Tier” architecture for simplicity and sustainable performance. Our architecture is composed of Presentation Tier, Controller Tier and Data Tier. Presentation Tier is responsible for front-end it is located on the client-side whereas Controller Tier and Data-Tier are located on the server-side. In addition to subsystem decomposition, hardware-software mapping, persistent data management, access control and security, global software control and boundary conditions are presented in the following subtopics.

## 3.2. Subsystem Decomposition

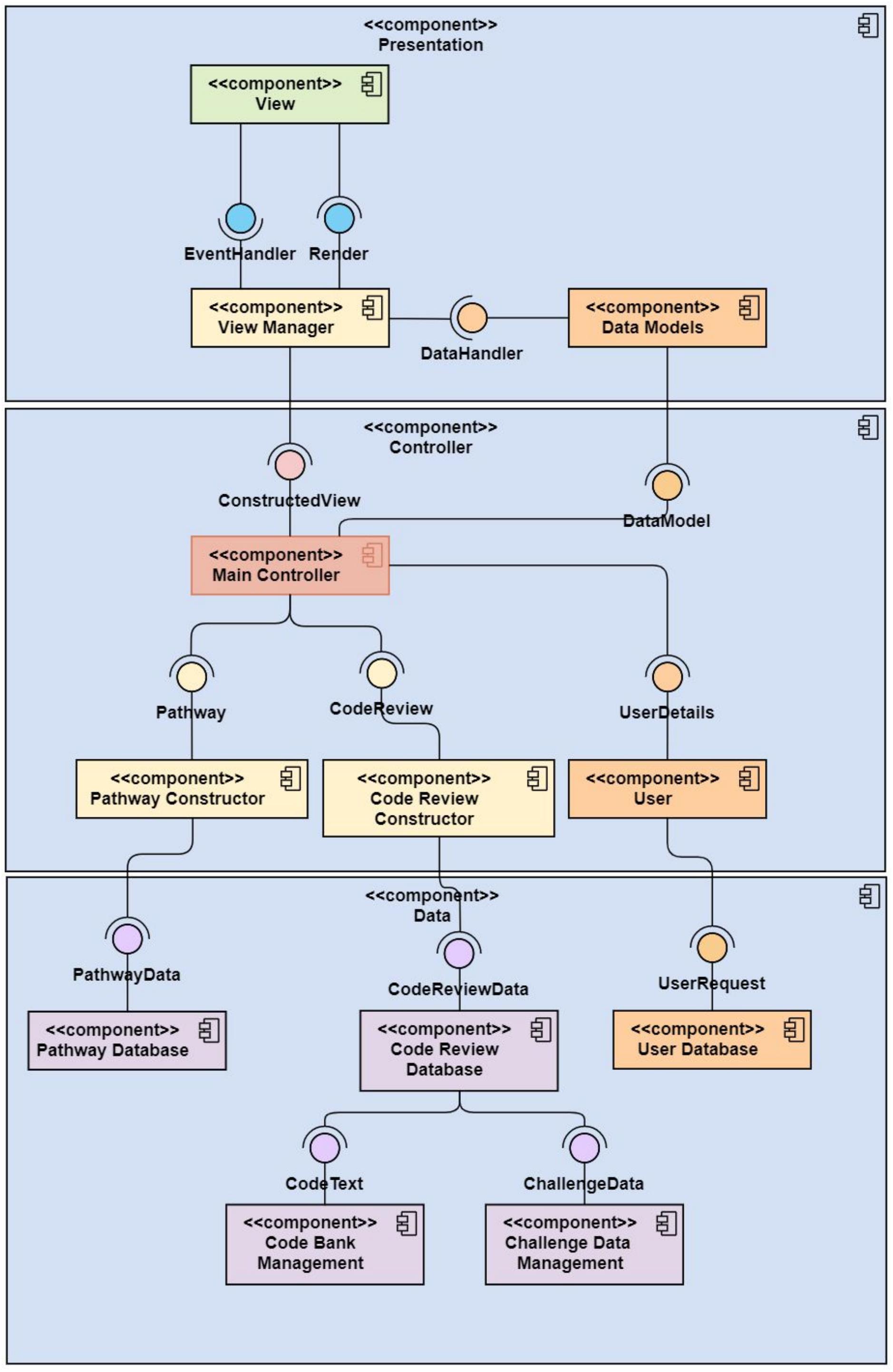

Figure 1: Subsystem Decomposition of Three Tier Architecture

Three-tier architecture is used by including Presentation Tier, Controller Tier and Data-Tier for decomposing the system. Presentation Tier is responsible from front-end and providing user interface. This side will be implemented with HTML5, CSS, Bootstrap and AngularJS technologies. Controller Tier is responsible for operating the core functionality of the system and will be implemented by Spring Java. Data Tier is responsible for data management of the system and it will be embedded by Spring Hibernate integration. REST API will be used for data services and Django will be used as a web server of the system.

## 3.3. Hardware/Software Mapping

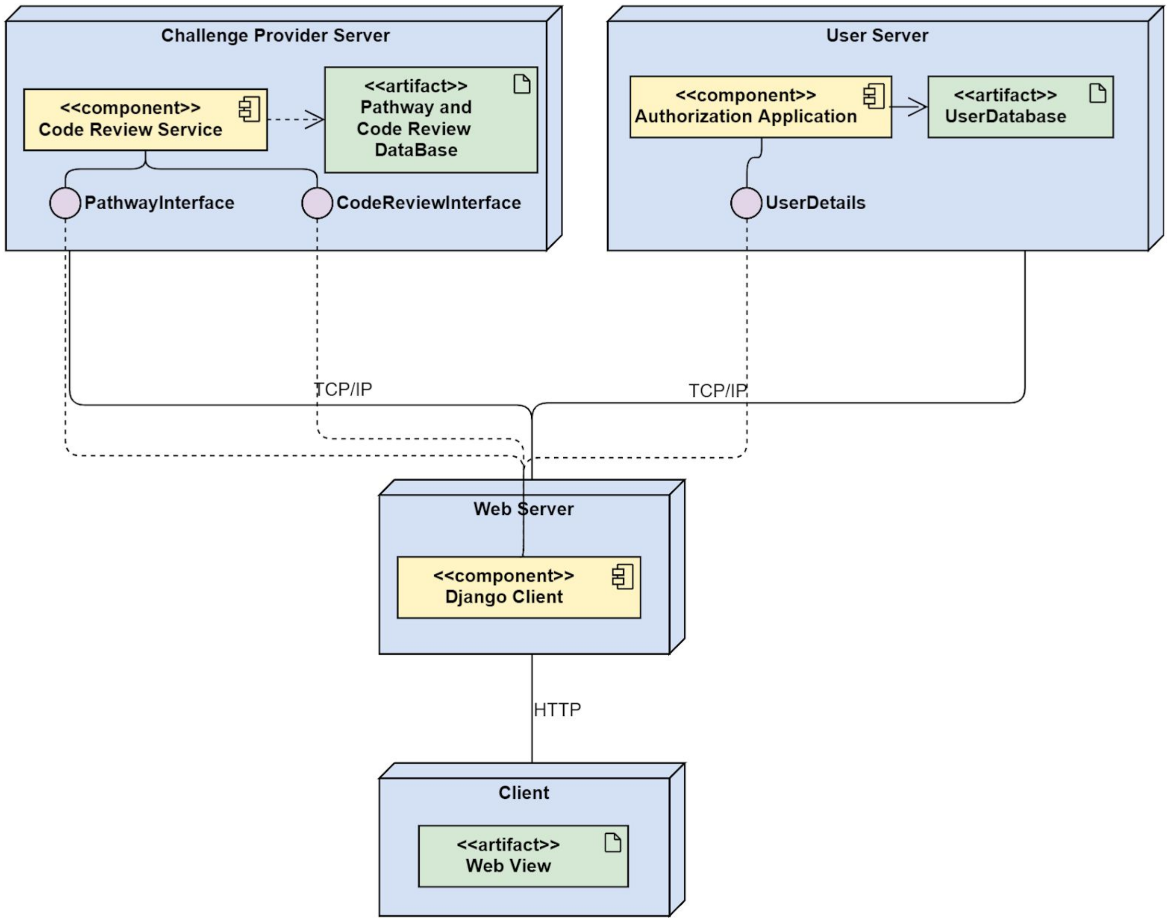
Challenge Provider Server
<<component>> Code Review Service
<<artifact>> Pathway and Code Review DataBase
PathwayInterface
CodeReviewInterface
User Server
<<component>> Authorization Application
<<artifact>> UserDatabase
UserDetails
TCP/IP
TCP/IP
Web Server
<<component>> Django Client
HTTP
Client
<<artifact>> Web View

Figure 2: Deployment Diagram

The system is distributed into 3 layers corresponding to Three Tier architecture. Client layer is responsible from user interface and deals with user inputs by creating corresponding requests. Web Server presents the Controller Tier and handles the requests between front-end and data layer. Data side consists of Challenge Provider Server and User Server. Challenge Provider Server stands mostly for gameplay purposes and User Server serves for managing the user data.

## 3.4. Persistent Data Management

CodeReviewGame is a social web application, more importantly, a game application for software developers. Most of the data is connected with the users. Users contain all the necessary data for their game experience. Data is not a simply save progression game feature. It has also social application data abilities such as profile management and ranking. Player (student) holds its account details and pathways which are assigned to them. Pathways hold code review challenges which are linked to textbase code. Both instructors and students holds the same data but instructors has additional features in the application.

## 3.5. Access Control and Security

The data provider of our product keeps all the information secure by HTTPS protocol and built-in encryption. Password is encoded with SHA256 on the client-side. Therefore, the real password is not delivered to servers which keeps real password safe. Internal systems such as the connection between data servers and the web server is secured by internal communication protocols. Since we adopted service base model on our product, data servers can be held outside of the web servers. In that case, security is maintained by firewalls that are provided by operating system companies.

Users can only access their own linked data. Those links can be created by instructors. By design, there are no admin or maintenance accounts in our system. Still, separation on access between instructor and student is maintained by authorization service. This service uses a built-in authorization plugin inside Django. This plugin uses the Oauth2.0 plugin for its authorization process.

## 3.6. Global Software Control

The application is a self-driven social game platform. In regular cases, it doesn't require control besides technical issues. Since our application is based on services, those services can be maintained and controlled separately. In an optimal scenario, those services are maintained by the product owner. For extreme social cases, system can be controlled by a feedback system between product owner and customer.

## 3.7. Boundary Conditions

### 3.7.1. Initialization

Our system will be initialized with a web browser application such as Google Chrome, Internet Explorer or Firefox. Users will encounter the main page if they are not logged in yet. Users can register and login on this page. An active internet connection is required in order to access and initiate the system.

### 3.7.2. Termination

Users can log out in order to terminate their session with the system. Cookies on client-side and Sessions on server side are held for player authorization. Users can remove their browser's catches in order to clean their session. System deletes any session information after 30 minutes from their last activity in the system.

### 3.7.3. Failure

The system won't be operational if there is a connection problem between client and server. Only snapshots of the application can be presented in such cases. Internal data services and web server connection problems will cause to close down new user logins and registers but users who are already connected to the web server may continue using the system since the web server holds login user data on its cache.

## 4. Subsystem Services

### 4.1. Presentation Tier

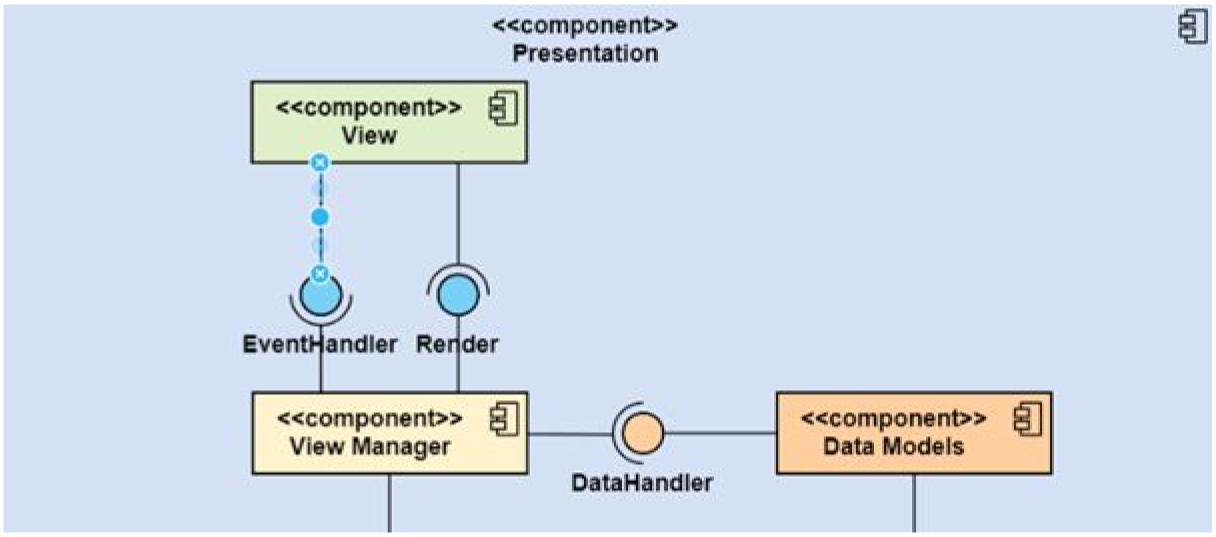

Figure 3: Presentation Tier

Presentation tier is responsible from the user-interface operations. In general concept, it gets the input from the user and transports to the controller tier. User-interface is accessible from web browsers. Presentation tier will have HTML5, CSS, Bootstrap and AngularJS technologies.

#### 4.1.1. View Component

- It is the component which is providing the front-end and user interface. View component gets the user input and transports it to the View Manager. Also requests come from the View Manager.

#### 4.1.2. View Manager Component

- View Manager is the manager of the Presentation tier. It receives user input from the View Component and data from the Data Models and also receives database requests.

- It modifies the screens according to requests created in Presentation tier and coming inputs from the Controller tier.
- Listens to the events of View Component via Event Handler.
- Take responds from the Controller tier.
- Updates the screen by Render.

### 4.1.3. Data Models Component

- Data Models Component is responsible for getting and dealing with any kind of data sent by the user input.

## 4.2. Controller Tier

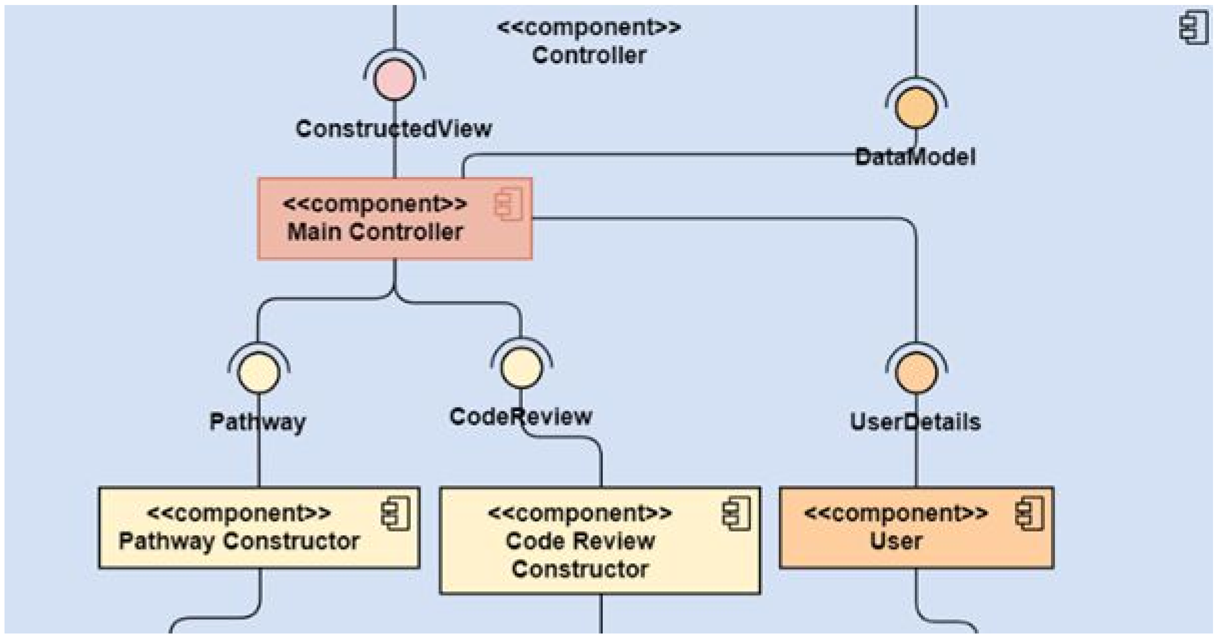

Figure 4: Controller Tier

The controller tier provides communication between Presentation and Data tiers. It is the main operator of the core functionality.

### 4.2.1. Main Controller Component

- Main Controller Component transports the data received from Presentation tier to the necessary components.
- For example in a login operation, when the user information (email and password) received from Constructed View, information passes to the

User component for verification. The user component verifies the given user information from Data-tier and Main Controller requests the user information from the User Component. When Main Controller takes the information from User Component, it sends it to View Manager for informing the user about the login process.

- The program presents two modes for code review challenges: following a pathway and single code review challenge. The main controller takes the requests for challenge mode from the input taken by Presentation Tier and transports it to the appropriate subcomponent for construction.

### 4.2.2. Pathway Constructor Component

- It is the subcomponent which is responsible for creating a pathway and operating data flow of it. The pathway is a simple structure that contains more than one code review challenge inside and code review challenges are ordered according to their difficulty levels. Pathway Constructor is invoked by the request from Main Controller.

### 4.2.3. Code Review Constructor Component

- It is the subcomponent which is responsible for constructing a code review challenge and operating data flow of it. It is invoked by the Main Controller.

### 4.2.4. User Component

- User Component is responsible for the user-related operations in the program. User related operations in the program are authentication, user scores, and user badges. Also, this component controls the data flow of users and provides communication between Data-tier for keeping user data updated.

## 4.3. Data Tier

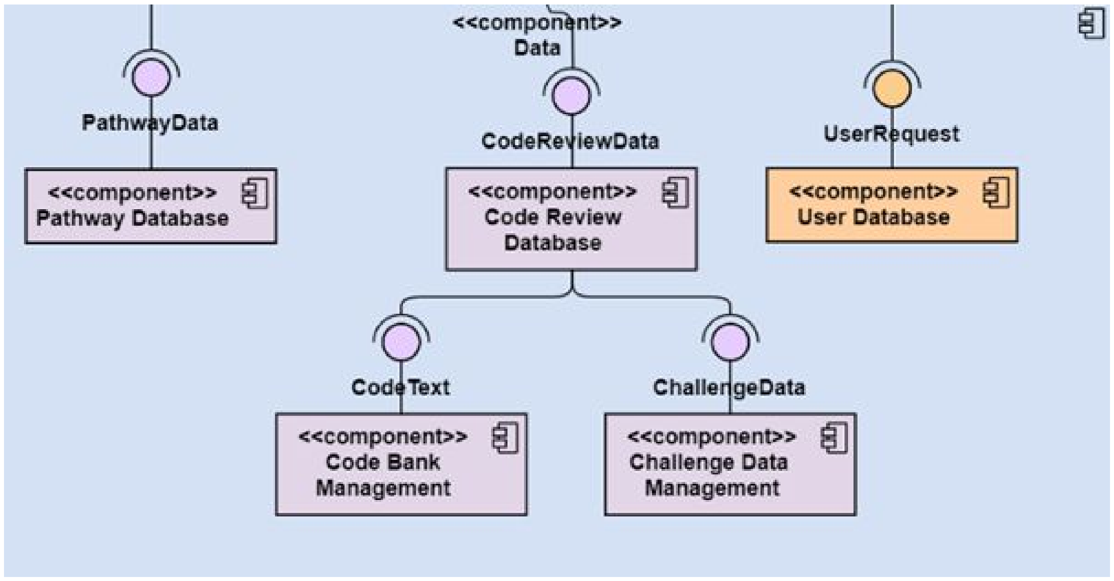

Figure 5: Data Tier

Data-tier is responsible for data handling of the system and data communication. Presentation tier does not have direct access to data tier. Data tier is just accessible by the requests from Controller tier.

### 4.3.1. Pathway Database Component

- This component is responsible for storing and providing Pathway data of the system. When a user creates or views a pathway, Pathway Database need to be called in order to provide data consistency.

### 4.3.2. Code Review Database Component

- This component is responsible for storing the whole data for code review challenges. It consists of two subcomponents: Code Bank Management Component and Challenge Data Management component. Code Bank Management is managing the code snippet data which are uploaded by the user (teacher). Challenge Data Management is responsible for storing and managing the defined defects for code snippets. Both of these subcomponents are holding the core functionality of data management for the gameplay side.

#### 4.3.3. User Database Component

- This component is responsible for all user-related data management. It stores the user information like username, e-mail, scores, rankings, badges, and shares these data with the system according to the coming request from the Controller tier.

## 5. New Knowledge Acquired and Learning Strategies Used

Django and bootstrap were learned which the former has a model template view architectural pattern, while the latter is a library that can be used in any platform, to create web pages using vast arrays of modern tools and features; such as bundled JavaScripts, browser compatibility, simple integration (just add as an external jar), extensive list of components, great grid system, base styling for most HTML elements, etc… Bootstrap is especially good for its ability to fit in screens dynamically.

In addition to these, we have acquired the knowledge of AngularJS. This will be used for web design as well, to further enhance the implementation. AngularJS has its own set of practical features. A number of them can be listed down as; templating, RESTful API handling, dependency injection, etc… There is another framework, Spring which can be used for this kind of work, however, AngularJS is more compatible with a user-friendly and easy to build environment.

## 6. 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.
- **Badges:** Achievements that user gains according to his/her performance in code review challenge.

## 7. References

[1] ”Code Review Guidelines”. [Online].

https://engineeringblog.yelp.com/2017/11/code-review-guidelines.html

[2] ”Lessons From Google: How Code Reviews Build Company Culture”. [Online]. Available:
https://blog.fullstory.com/what-we-learned-from-google-code-reviews-arent-just-for-catching-bugs/

[3] ”What a Good Code Review Looks Like”. [Online]. Available:

https://www.codewall.co.uk/what-a-good-code-review-looks-like/

[4] ”Speed of Code Reviews”. [Online]. Available:
https://google.github.io/eng-practices/review/reviewer/speed.html