



Bilkent University

Department of Computer Engineering

# Senior Design Project

*The Game – Code Review*

## Final Report

Burak Erkiş	21501035
Mert Sezer	21400246
Okan Şen	21202377
Berk Yıldız	21502040

Supervisor: Halil Altay Güvenir

Jury Members: Shervin Rahimzadeh Arashloo, Can Alkan

Final Report

May 27, 2020 (Last Update: May 27, 2020)

This report is submitted to the Department of Computer Engineering of Bilkent University in partial fulfillment of the requirements of the Senior Design Project course CS491/2.

## Table of Contents

<b>1. Introduction</b>	<b>4</b>
<b>2. Requirements Details</b>	<b>4</b>
<b>2.1. Functional Requirements</b>	<b>4</b>
<b>2.2. Non-functional Requirements</b>	<b>6</b>
<b>2.3. Pseudo Requirements</b>	<b>6</b>
<b>3. Final Architecture and Design Details</b>	<b>6</b>
<b>3.1. Software Architecture</b>	<b>6</b>
<b>3.1.1. Overview</b>	<b>6</b>
<b>3.1.2. Subsystem Decomposition</b>	<b>7</b>
<b>3.1.3. Hardware/Software Mapping</b>	<b>8</b>
<b>3.2. Subsystem Services</b>	<b>9</b>
<b>3.2.1. Presentation Tier</b>	<b>9</b>
<b>3.2.2. Controller Tier</b>	<b>10</b>
<b>3.2.3. Data Tier</b>	<b>11</b>
<b>3.3. User Interfaces</b>	<b>12</b>
<b>3.3.1. Login</b>	<b>12</b>
<b>3.3.2. Register</b>	<b>13</b>
<b>3.3.3. Create Class (Teacher)</b>	<b>13</b>
<b>3.3.4. Code Editor</b>	<b>14</b>
<b>3.3.5. Pathway (Student)</b>	<b>14</b>
<b>4. Development/Implementation Details</b>	<b>15</b>
<b>4.1. Main Features of Implementation</b>	<b>15</b>
<b>4.2. Adapted Technologies</b>	<b>17</b>
<b>4.3. Implementation Approach</b>	<b>17</b>
<b>4.3.1. Backend Implementation</b>	<b>17</b>
<b>4.3.2. Frontend Implementation</b>	<b>19</b>
<b>4.3.3. Data Models</b>	<b>20</b>
<b>4.4. Libraries Used</b>	<b>21</b>

4.4.1. Frontend Libraries .....	21
4.4.2. Backend Libraries .....	21
4.5. Summary of Technologies and Tools Used .....	22
5. Testing Details .....	22
5.1. Test Scenarios for Student Role .....	23
5.1.1. Login .....	23
5.1.2. Register .....	23
5.1.3. Profile.....	23
5.1.4. Gameplay.....	23
5.2. Test Scenarios for Teacher Role.....	24
6. Maintenance Plan and Details .....	24
7. Other Project Elements .....	25
7.1. Consideration of Various Factors .....	25
7.2. Ethics and Professional Responsibilities .....	27
7.3. Judgments and Impacts to Various Contexts .....	28
7.4. Teamwork and Peer Contribution.....	30
7.5. Project Plan Observed and Objectives Met .....	30
7.6. New Knowledge Acquired and Learning Strategies Used.....	31
8. Conclusion and Future Work .....	32
9. Glossary.....	32
10. References.....	33

# 1. Introduction

Code review is an important necessity for software development however is seen as a bothering task for most of the software developers. One of the reasons for that is, usually, code review is not taught as a requirement in the development process to students or if it taught as a requirement, the practical approach is missing toward the code review. To change this approach, we are providing a code review game that will gamify the code review especially for the students who are learning software development. Our goal is to adopt the habit of code reviewing to the coders of the future starting from the first steps of education.

Writing code or reviewing one can be quite tedious and energy-draining at times and this tiresome process can result in revisions or codes to be faulty and even inefficient. The time these actions require grows exponentially into tremendous amounts as the complexity of the work increases. In order to soften the burden of these works, The Game comes up with several innovative ideas. The classic, repetitive methods can be gruesome and cause disinterest within coders which we want to avoid. The Game focuses on creating a more appealing environment to learn about reviewing codes using game modes such as; challenges within classes or in a single form, defect detection. There are teacher and student user types. Teachers can create and post new challenges thorough pathways and codes to be reviewed, while the students will have to work their way around these challenges and find the defects in these faulty codes. The Game can work as a social/professional platform bringing many likeminded people together, helping them to find friends to discuss code related topics or to find group members for their projects.

## 2. Requirements Details

### 2.1. Functional Requirements

- The Game includes two actors:
  - Teacher – the one who posts the challenges
  - Student – the one who solves the challenges
- All actors should login to the game by their saved e-mail addresses and passwords. The accounts which are not logged in are not allowed to use facilities of the game
- The Game has seven main features:

- Pinpoint Defects (Student)

The goal is to mark defects and to do that the player has to follow some steps. The player has to be sure that there is some code selected in the code area. The player can click one defect buttons to mark the excerpt as a defect. Possible defect types are Functionality, Complexity, Naming, Comments, Style and Consistency. In the code, the excerpt will be highlighted and the defect will be added to the list in the sidebar, having a border with a certain color according to the selected severity and the description equal to the selected type.

- Pick & Unpick Defects (Student)

A player simply picks or unpicks a defect according to the available option for the defect that depends on its current state. If the defect is picked, it will be colored on the list and marked on the code in a bright color, with the available option being to unpick. If it's unpicked, it will marked with a darker color on the code, while the available option is to pick the defect.

- Create Pathway (Teacher)

Teacher creates pathways which include several challenges in different levels.

Teacher can assign created pathways to classrooms. Students won't be able skip a level without solving the current one.

- Teacher can upload the code script or create pathway by indicating its programming language and enter the defects into the system.
- Rankings and statistics can be shown in Scoreboard and winner of the challenge will be stated at the end of each challenge.
- Teacher is able to review pathways and code scripts, students are able to preview previous code review results and class rankings.
- Students can enroll a class from a link provided by teacher.

## 2.2. Non-functional Requirements

- **Portability:** The Game is a web program and going to work in all browsers with any operating system on desktop.
- **Usability:** The Game has a simple user-friendly interface. A user is able to use all facilities in his/her first meet with the program by our basic navigational path.
- **Privacy:** All users should login to system for using the facilities. All personal data, e-mail addresses and password will be protected and will never share with any third parties.
- **Availability:** There is no need of extra installation to run program. Program is accessible from all type of internet browsers.
- **Scalability:** System is able to respond several requests because especially in class work, several requests will arrive concurrently.
- **Performance:** System responds to user in acceptable amount of time (less than a second) for any sort of request.

## 2.3. Pseudo Requirements

- The Game is a desktop web program. Internet is needed to access.
- Frontend is implemented with AngularJS.
- Backend is implemented with Django Rest Framework.
- MongoDB is used for database.

# 3. Final Architecture and Design Details

## 3.1. Software Architecture

### 3.1.1. 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.

### 3.1.2. Subsystem Decomposition

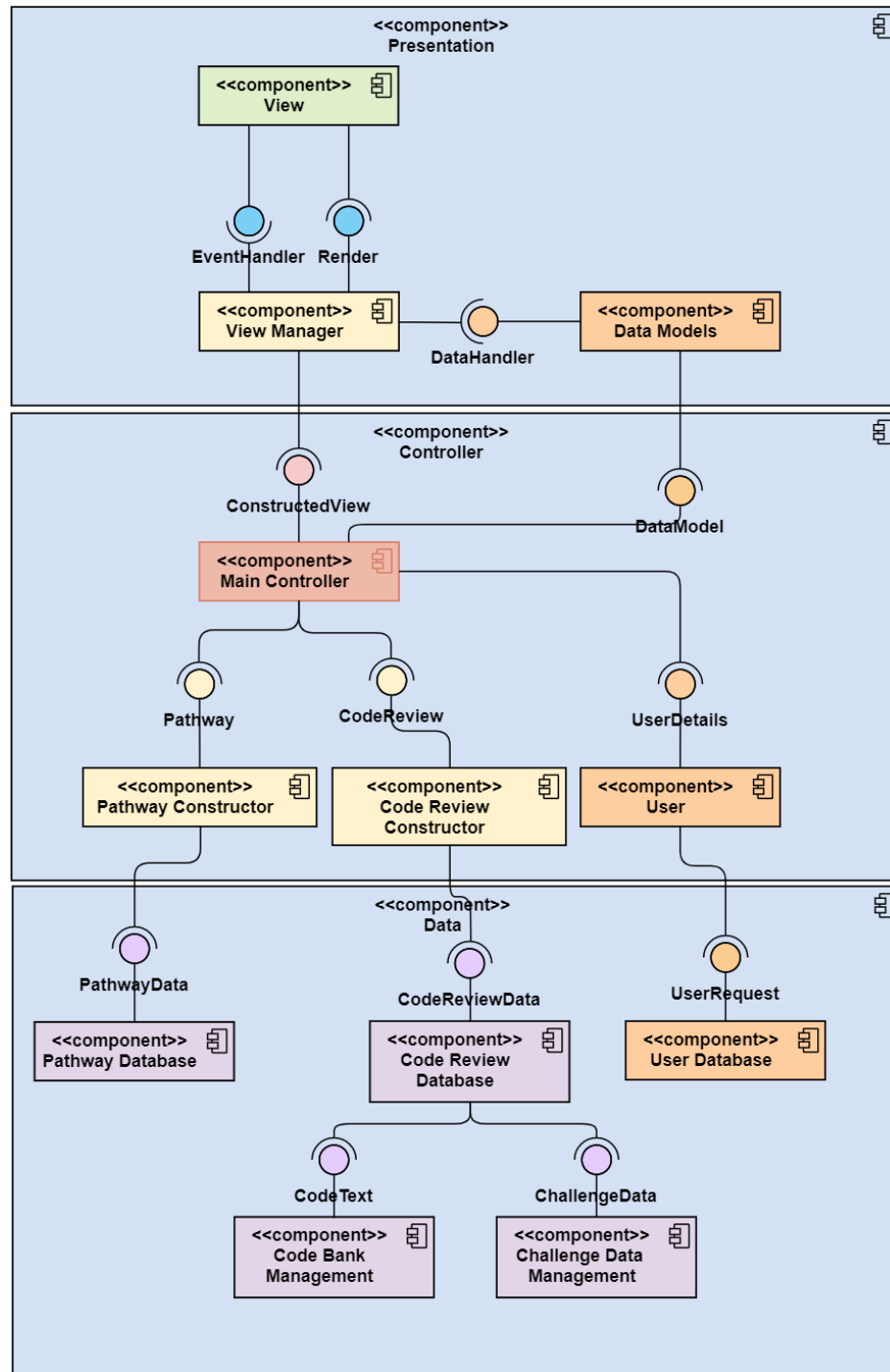


Figure 1: Subsystem Decomposition of Three Tier Architecture

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 is implemented with AngularJS technologies. Controller Tier is responsible for operating the core functionality of the system and is

implemented by Django. Data Tier is responsible for data management of the system and it is embedded by Django REST integration.

### 3.1.3. Hardware/Software Mapping

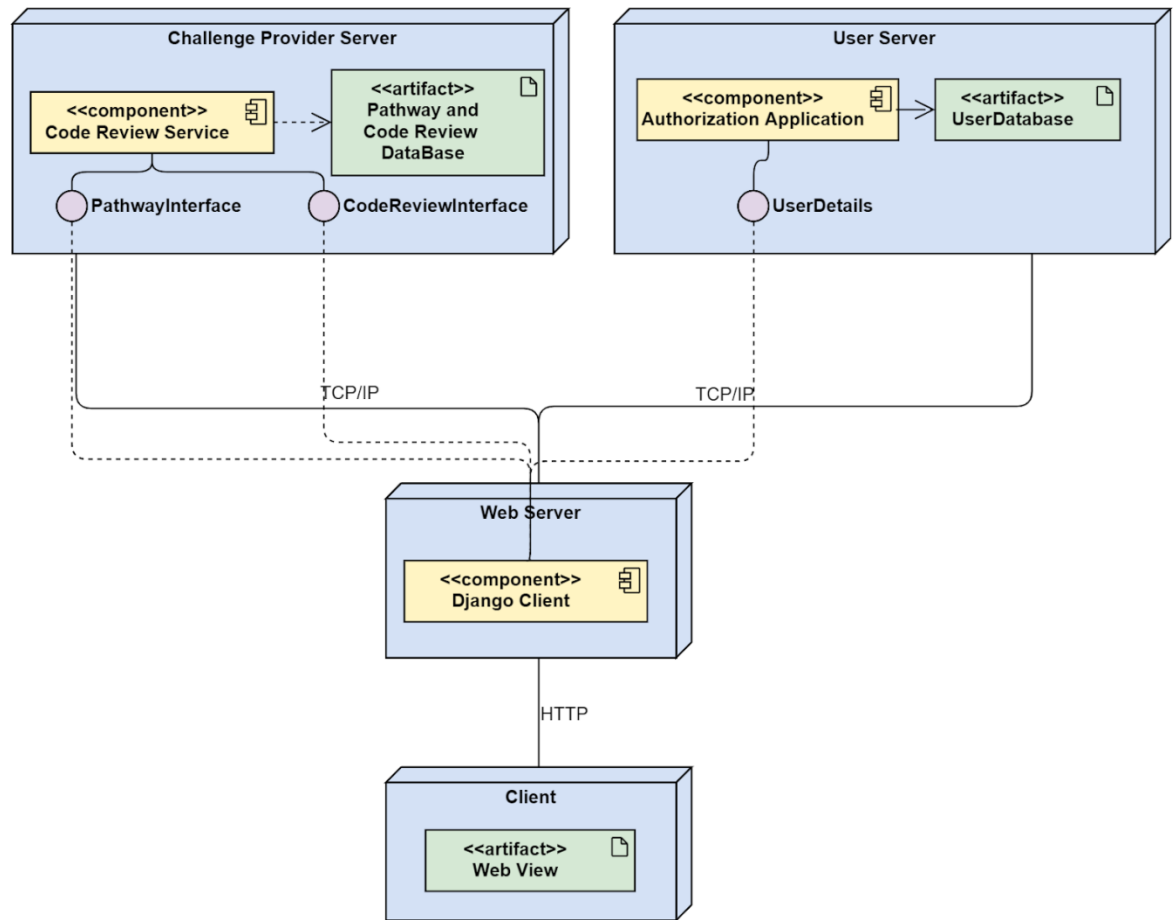


Figure 2: Deployment Diagram

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.



## 3.2. Subsystem Services

### 3.2.1. Presentation Tier

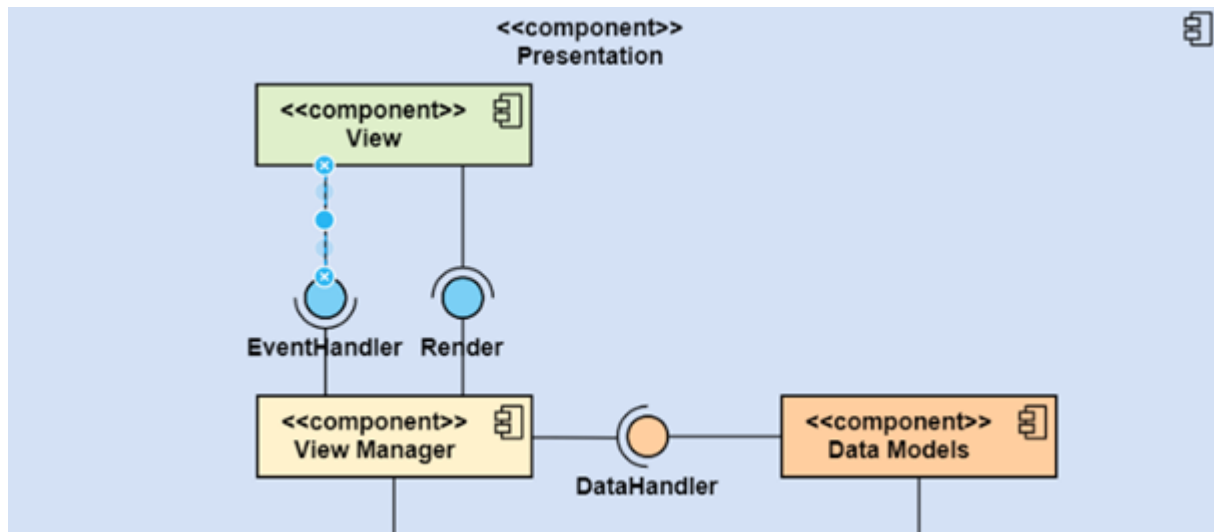


Figure 3: Presentation Tier

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 is implemented by AngularJS technologies.

#### 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.

#### 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.

### Data Models Component

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

### 3.2.2. Controller Tier

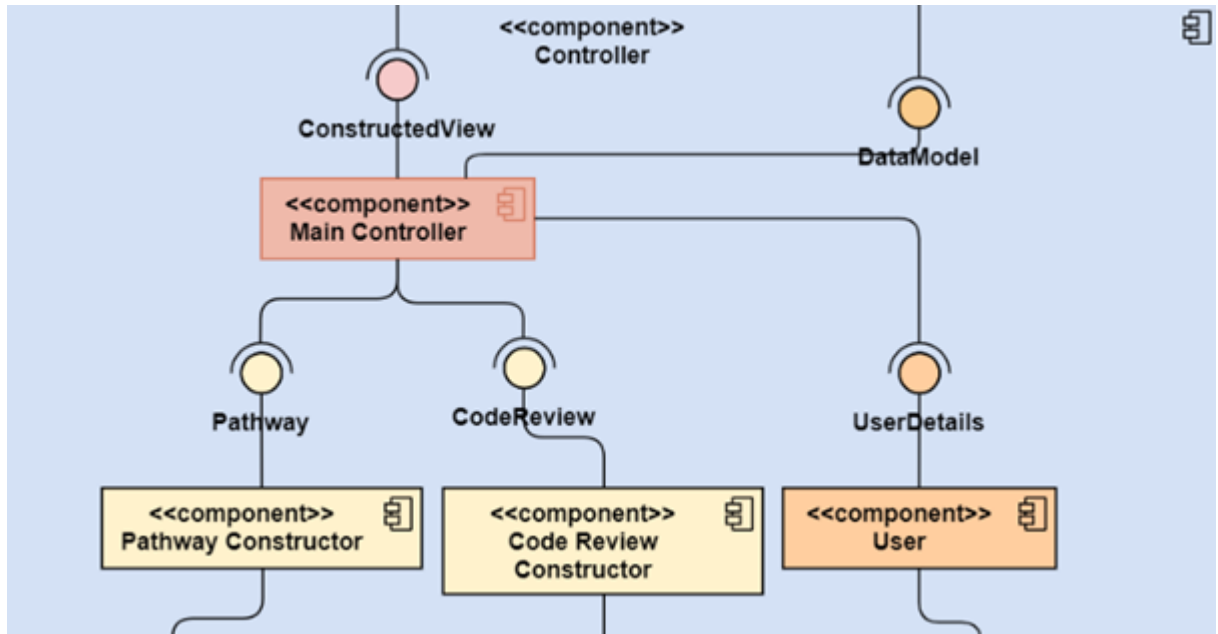


Figure 4: Controller Tier

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

### 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.

### 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.

### 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.

### 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.

### 3.2.3. Data Tier

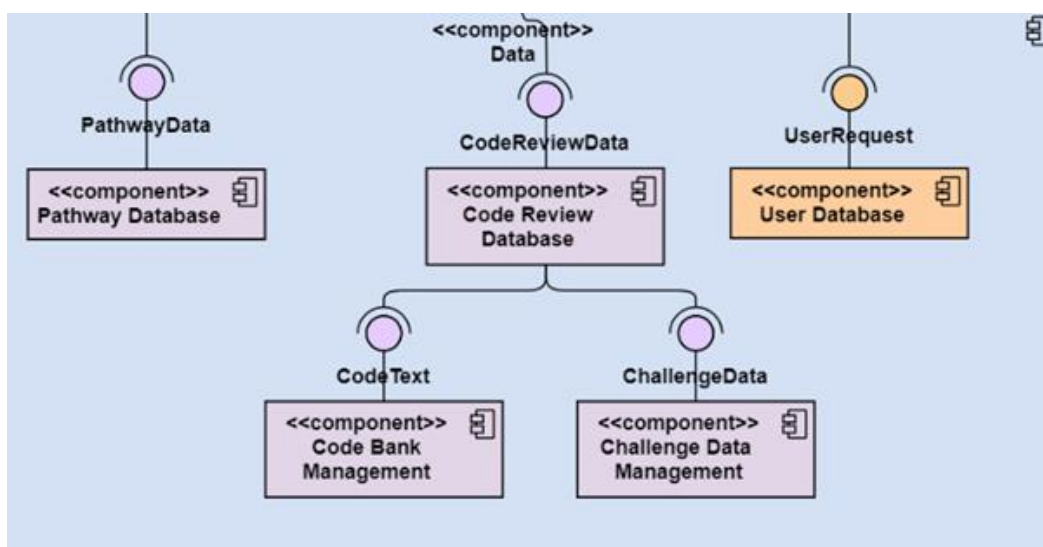


Figure 5: Data Tier

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.

### **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.

### **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.

### **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.

## **3.3. User Interfaces**

### **3.3.1. Login**

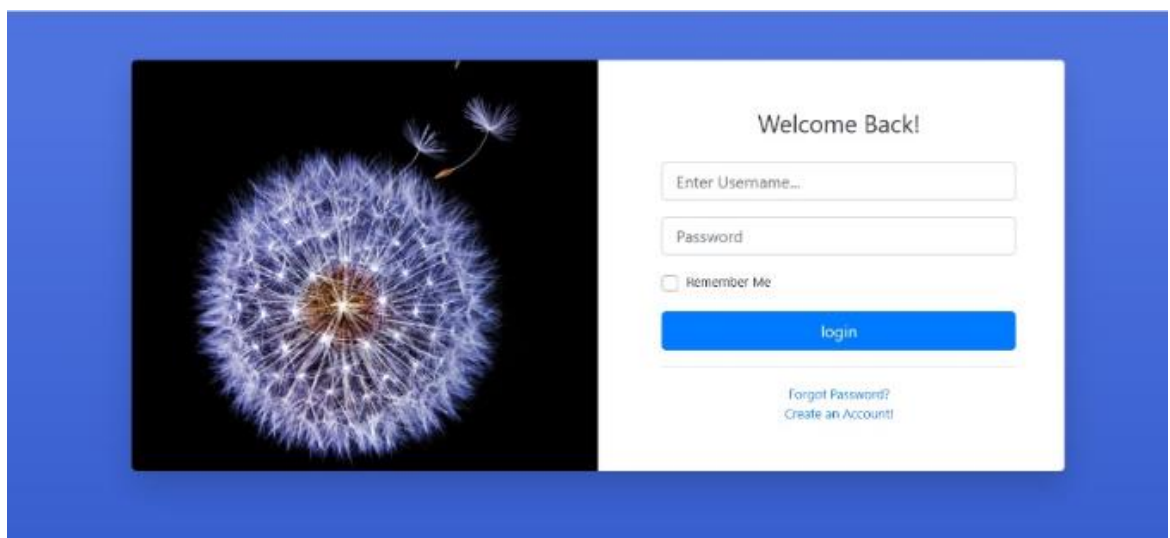


Figure 6: Login Page

Both student and teacher have to login to system with their username and password for using the application. Unregistered users are not allowed to enter the system

### 3.3.2. Register

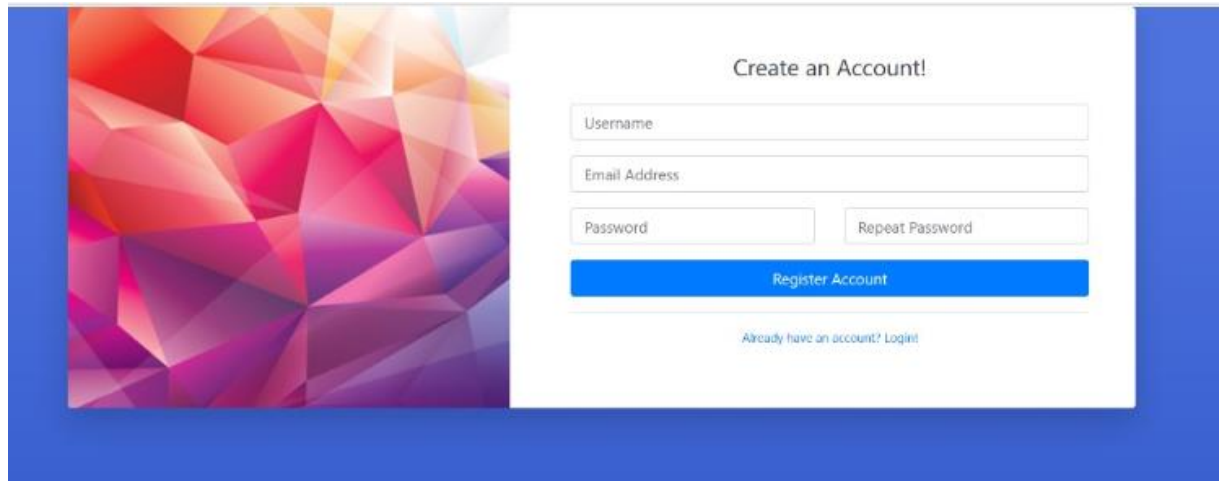
A screenshot of a web application's registration page. The page has a blue header and a white main content area. On the left, there is a decorative graphic with a colorful, low-poly geometric pattern in shades of red, orange, and purple. The main content area is titled "Create an Account!". Below the title, there are four input fields: "Username", "Email Address", "Password", and "Repeat Password". A blue button labeled "Register Account" is positioned below these fields. At the bottom of the form, there is a link that says "Already have an account? Login!".

Figure 7: Register Page

User who do not have account have to register to system with his/her email. Also user has to determine a unique username and regular password.

### 3.3.3. Create Class (Teacher)

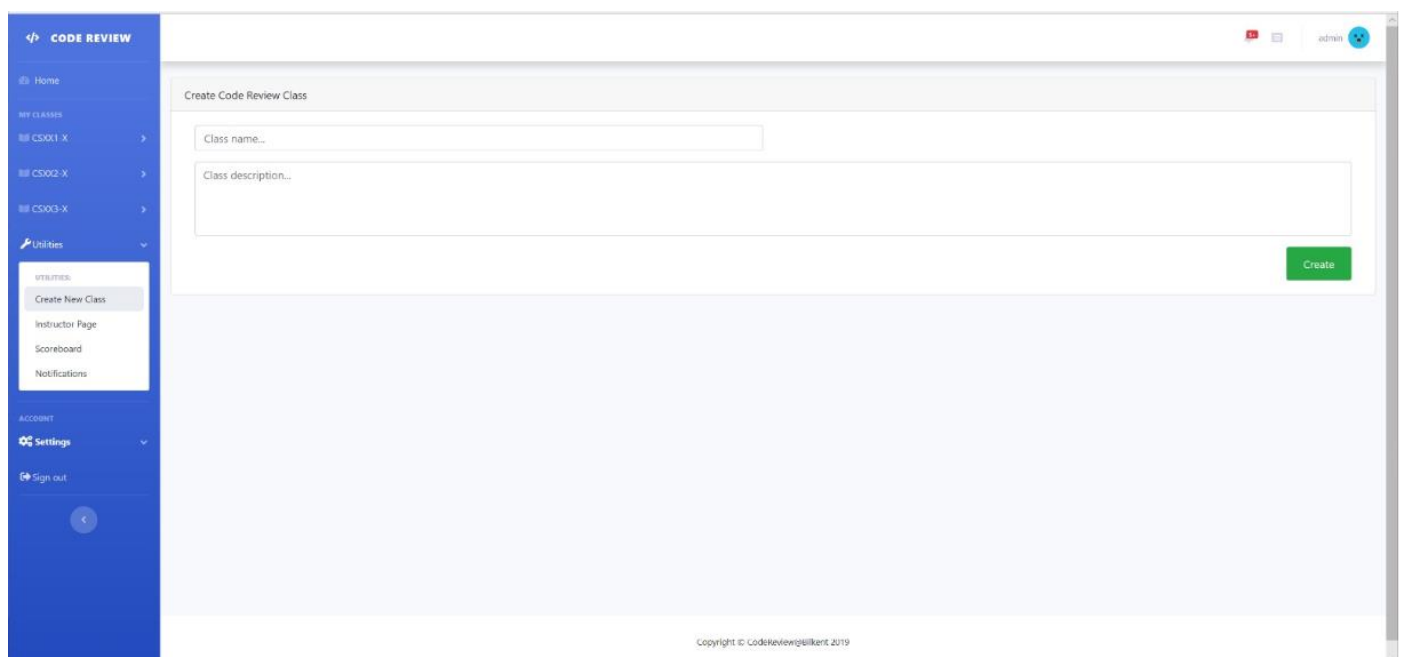
A screenshot of a web application's "Create Class" page. The page has a blue sidebar on the left and a white main content area. The sidebar contains a "CODE REVIEW" header, a "Home" link, a "MY CLASSES" section with links for "CS011-X", "CS012-X", and "CS013-X", a "Utilities" section with links for "Create New Class", "Instructor Page", "Scoreboard", and "Notifications", and an "ACCOUNT" section with links for "Settings" and "Sign out". The main content area is titled "Create Code Review Class". It contains two input fields: "Class name..." and "Class description...". A green button labeled "Create" is located at the bottom right of the form. The footer of the page contains the text "Copyright © CodeReviewGillKent 2019".

Figure 8: Create Class Page

Teacher can create a class easily by just providing class name and class description.

### 3.3.4. Code Editor

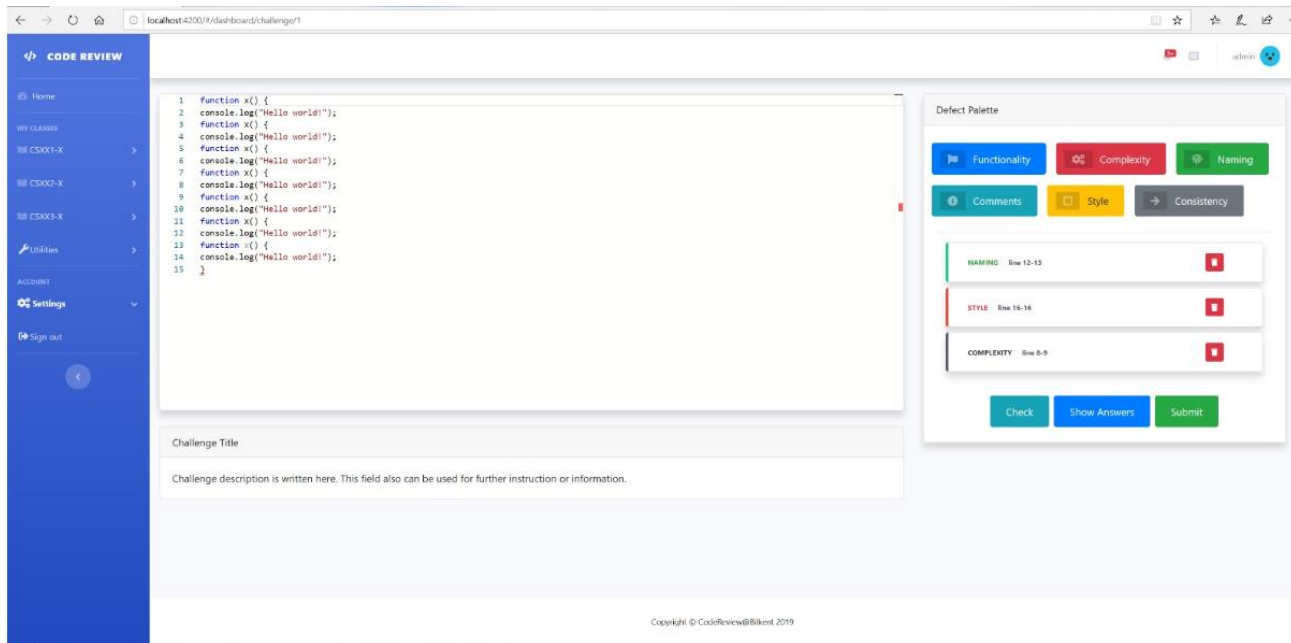


Figure 9: Code Editor

This is the main gameplay screen of the program. Middle of the page is allocated for editor and right side allocated for defect selection. Student should select lines of code from the editor and select a defect type from the defect palette on right side. Selected defects are also listed in the right hand side of the screen.

### 3.3.5. Pathway (Student)

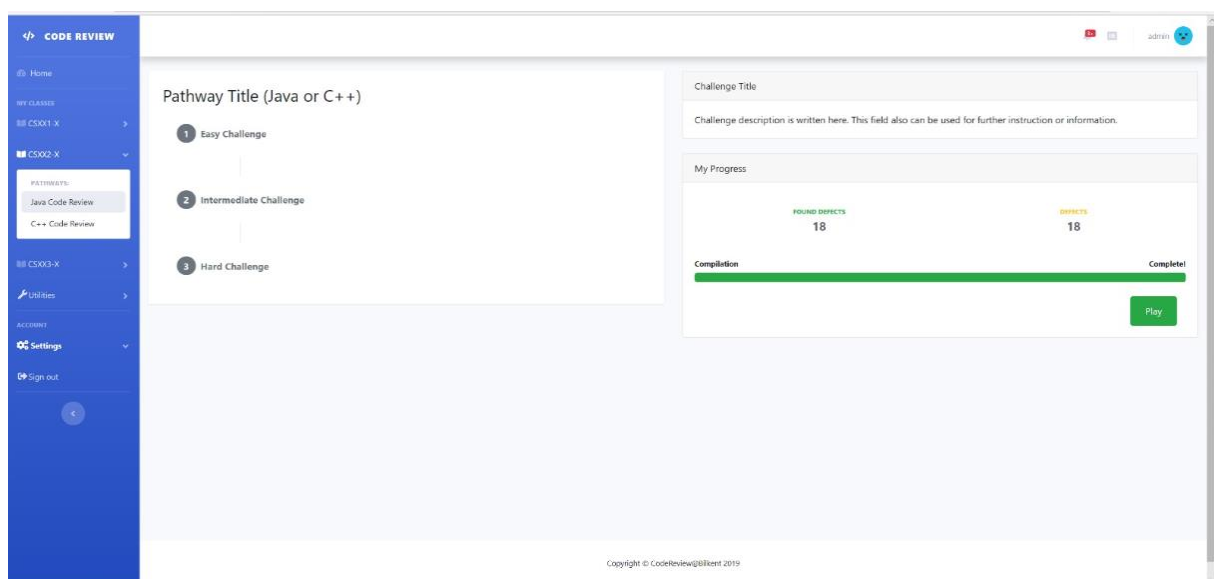


Figure 10: Pathway Screen

Pathway is simply a series of challenges. Student can enter an assigned pathway from this page and start to solve challenges one by one. Clicking on Play button at right-hand side is enough for starting to adventure.

## 4. Development/Implementation Details

### 4.1. Main Features of Implementation

- **Security**

Security is one of the priorities in the project. JSON Web Tokens is implemented in project for user authentication. According to JWT, client sends a login request to the server side with the username (or mail) and password on the login page. The server checks the incoming information with a database query. If it is invalid, authentication error returns, if it is valid, creates a token with user information and a predefined secret key. This token sends to the client side in the 'Header' section of the HTTP request. Client keeps incoming token information in local storage. Then, in all requests (get, post, put, delete...) requesting authorization, it sends this token information in Header information. In every request, the server verifies the token and controls whether the user can access it. If access is accepted, the data flow continues normally. Returns '403 Forbidden' to client side if not accept.

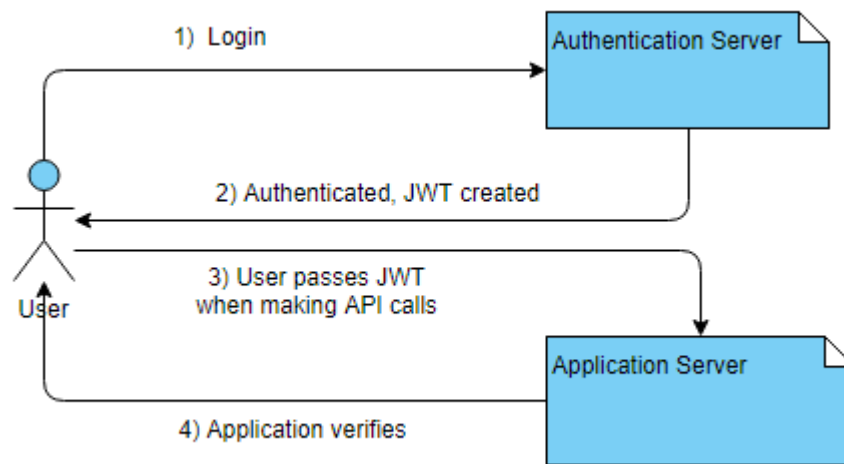


Figure 11: Flow of JWT

- **Responsive Web Design**

With the diversity of electronic device usage, portability has gained importance in terms of user experience. Website of Code Review Game is responsive and suitable for all-sized and type electronic devices including tablets, computers and mobile devices by implementation of Bootstrap in frontend development.



Figure 12: Responsive Structure of Bootstrap [1]

- **Single Web Application**

Single Page Application is a web application where routing is done by the client-side by JavaScript, not by the backend. In summary, there is only one index page on our website, and the pages are changed by the router and displayed to the user on the client side. While doing this, we get help from web components. The router shows the components you set for the path to the user.

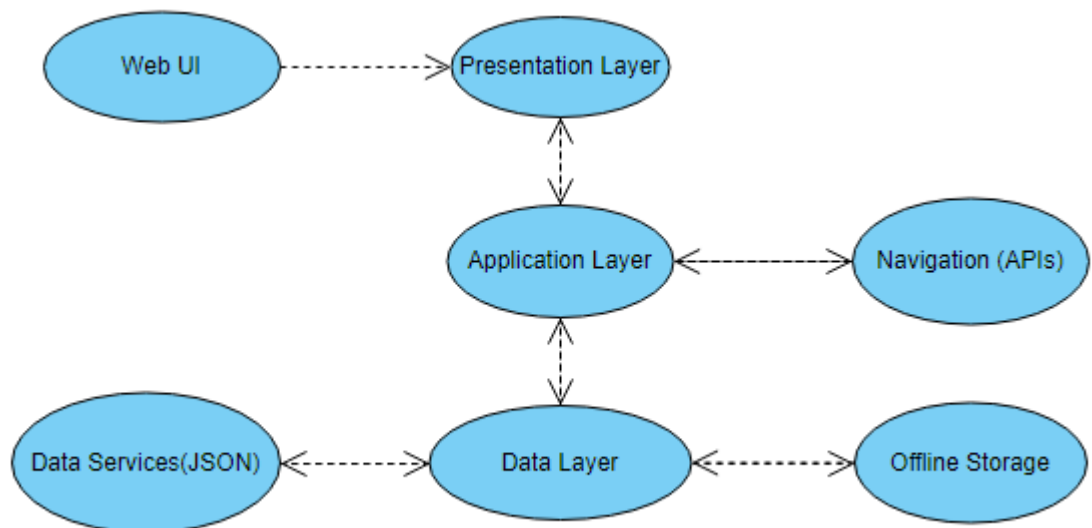


Figure 13: Architecture of SPA



## 4.2. Adapted Technologies

- **MongoDB**

MongoDB is a cross-platform document-oriented database program. Classified as a NoSQL database program, MongoDB uses JSON-like documents with schema [2].

- **Django Rest Framework**

It is web browsable version of API and it offers an option of returning JSON. It provides powerful model serialization; display data using standard function based views, or get granular with powerful class based views for more complex functionality. We preferred using this framework in our project; because it is simple to use; it creates, views and revokes API keys via the admin site, or use built-in helpers to create API keys programmatically and it is customizable; in other words it satisfies specific business requirements by building your own customized API key models, permission classes and admin panels [3].

- **AngularJS**

It is a Javascript-based open-source front-end framework that is mostly used for the development of single page web applications and with AngularJS, we created a dynamic web content through using HTML as the template language; we extended HTML with new attributes. Also because AngularJS is fully extensible and works well with other libraries, features can be modified for development workflow and feature needs, we decided to use this language in our project [4].

## 4.3. Implementation Approach

### 4.3.1. Backend Implementation

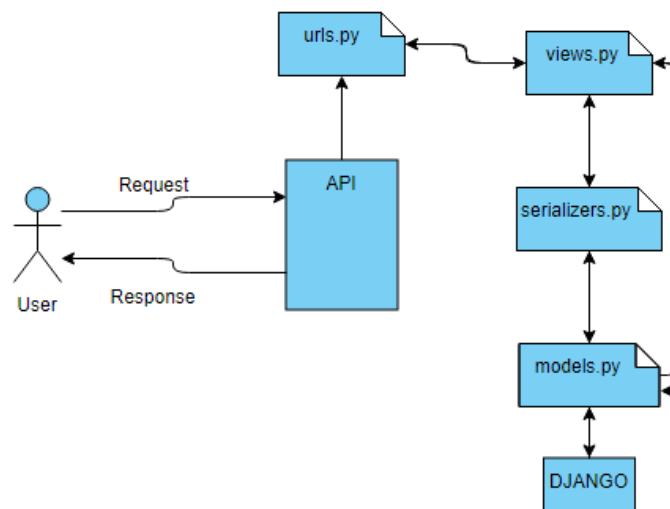


Figure 14: Backend Implementation

- **urls.py**

urls.py is the file which controls the operation of both the local Django packages and the packages created by other developers within the Django project, in which we set the routing rules regarding our Django project and the applications.

- **views.py**

As a result of the requests made to the APIs we have created with Django Rest Framework, we need to adjust which model, what type of request and what to respond. Instead of performing these operations manually, DRF performs all of these things with our viewsets class. We are using the `ModelViewSet` class in its simplest form. This class creates ready-made functions for operations on complex model structures. Functions such as `list()`, `retrieve()`, `create()`, `update()`, `destroy()` are the default structures we can access within this class. If we wish, we can arrange these methods according to ourselves. We can define more than one View for a single serialized model according to the scale of the application we make and the changes we will make on the data.

In this scope five ViewSets are presented:

- **UserProfileViewSet:** Includes view interactions of user profile.
- **ClassItemViewSet:** Includes view interactions of classes.
- **EnrollClassView:** Includes view interactions for enrolling into a class.
- **PathwayView:** Includes view interactions of a pathway.
- **ChallengeView:** Includes view interactions of challenges.

- **serializers.py**

This converts the data in classes, datasets and model classes into JSON for easy processing. In this scope there are six serializer classes defined.

- `UserProfileSerializer`
- `ClassItemSerializer`
- `PathwayItemSerializer`
- `ChallengeItemSerializer`
- `ChallengeProgressSerializer`
- `StudentProgressSerializer`

- **models.py**

Each table in the database is Python objects derived from the basic Model class, located in models.py. Each variable (field name) in this class provides the creation of columns and tables (For example ManyToMany fields) in the database table. Details of models will be given in 4.3.3 Data section.

#### 4.3.2. Frontend Implementation

- **Component Classes:**

- **login.component.ts:** Includes visible components of login page.
- **login.components.html:** Skeleton of the login page.
- **signup.components.ts:** Includes visible components of sign up page.
- **signup.components.html:** Skeleton of the sign up page.
- **enroll.component.ts:** Includes visible components of enroll to class page.
- **enroll.component.html:** Skeleton of enroll to class page.
- **dashboard.component.ts:** Includes visible components of dashboard.
- **dashboard.component.html:** Skeleton of the dashboard.
- **challenge.component.ts:** Includes visible components of challenge page.
- **challenge.component.html:** Skeleton of challenge page.
- **create-class.component.ts:** Includes visible components of create class page.
- **create-class.component.html:** Skeleton of create class page.
- **pathway.component.ts:** Includes visible components of pathway page.
- **pathway.component.html:** Skeleton of pathway page.

- **Service Classes:**

- **auth.service.ts:** Connects frontend and backend in terms data communication for authentication purposes.
- **api.service.ts:** Main service which provides data communication between frontend and backend in whole program.

- **Module Classes:**

- **dashboard.module.ts:** Includes sub-modules and sub-pages of dashboard and sustains swapping between sub-pages.

- **app.module.ts:** Main module class of the website. All sub-modules like LoginComponent, SignupComponent, EnrollComponent are included in this file.
- **app-routing.module.ts:** Provides navigation between sub-modules.

#### 4.3.3. Data Models

- **User Profile**

- id
- user
- profileName
- profileImage

- **Class Item**

- id
- title
- description
- user
- students
- instructors
- token: enrollment key for students

- **Challenge Item**

- id
- order
- title
- description
- challengeKey
- challengeText: code script to be solved

- **Challenge Progress**

- id
- student
- score
- challengeKey

- **Pathway Item**
  - id
  - order
  - classOwner
  - challenges
  - pathwayType: preferred programming language
  - title
  - description
  - challengeAmount: number of defined challenges
- **Student Progress**
  - id
  - student
  - pathway
  - challengeProgresses

## 4.4. Libraries Used

### 4.4.1. Frontend Libraries

- **ngx-monaco-editor:** Code editor
- **@fortawesome/angular-fontawesome:** Appearing emoji and images
- **Bootstrap:** Responsive web design and images
- **Jquery:** Instant page changes
- **chart.js:** Tables and schemes
- **jquery.easing:** Effects and smoother interfaces
- **bs-stepper:** Listing visual of pathway

### 4.4.2. Backend Libraries

- **JSON Web Tokens:** Encoder to generate secure messages after login
- **alluth:** Mail verification for register process
- **Django Rest Framework:** Backend server

#### 4.5. Summary of Technologies and Tools Used

- **HTTP Server:** Django
- **Rest Server:** Django Rest Framework
- **Single Site Page Engine:** Angular
- **Database:** MongoDB
- **Authentication:** JSON Web Tokens
- **Data Type for Backend & Frontend Communication:** JSON
- **Code Editor:** Monaco
- **Packager:** Webkit
- **HTML Stylizer:** Bootstrap
- **Theme:** SB Admin 2 [5] (Only material design is used)

### 5. Testing Details

The detailed design of Code Review Game is divided into two parts: back-end and front-end. The reasons behind this separation are that modularity increases and different people can simultaneously work on different parts of the project without messing up with other person's work and different people that are working on different parts can test and deploy their code without waiting for another person's completion. The con of this separation is that independent testing, build and deployment strategies for the application are separated by front-end and back-end and this takes time and resource. It is also important to ensure that front-end side is not fully separated from the back-end side and the database connection should be established. For instance, list of pathway nodes after the connection is established and the user object data should be checked to ensure front-end and back-end connection. The back-end design test should be done in a way that three main elements that are database, APIs and servers should be validated. For instance, for validating the API, once the installation is complete, the respective folder must be searched to check the files/elements having their target folders. The front-end design test should be done in a way that all team members must be sure that all components like buttons, filters are working as intended. For instance, every user click should work to deliver a nice experience to the user. In terms of unit and integration tests, we used Karma Test Runner of AngularJS[6] for the frontend to meet the test conditions we set. For the unit tests of backend, Django's own test engine is used.

A comprehensive testing process is one of the important factors to verify the smoothness and quality of the product. Therefore, the test phase of the project was taken seriously and a systematic test process was provided. Automation was preferred in addition to unit and integration testing in order to test main flow and use case scenarios. The automation process was not difficult to set up as the product is already web-based. In this context, Selenium was used for automation.

When choosing test cases, the priority was to reach maximum coverage with as few scenarios as possible. The combinatorial testing approach was followed by considering prioritizes aspects on testing process and taking into account that errors can arise from the interaction of multiple parameters. In this context, some of the important test scenarios that have direct effects on main flow are listed below.

## **5.1. Test Scenarios for Student Role**

### **5.1.1. Login**

- Student enters valid username and password and login. Verify if student can reach homepage.
- Student enters valid username and invalid password. Verify if system does not allow login process.

### **5.1.2. Register**

- Student enters valid input for all given fields. Verify if student can register to system.
- Student leaves username input blank and fills remaining fields. Verify if correct alert message shows up.

### **5.1.3. Profile**

- Student login to system. Clicks Profile and joins a class. Verify if student joined the classroom.
- Student clicks Profile on homepage. Clicks Edit Profile and updates password. Verify if update is done successfully.

### **5.1.4. Gameplay**

- Student joins a code review session. Pins the defects and submits solution. Verify if student's score calculated correct and he or she ranked successfully.

## 5.2. Test Scenarios for Teacher Role

- Teacher uploads a code review and assigns code review challenge for class. Verify if students can join the challenge and pin the defects.
- Teacher creates a pathway and assigns code review challenges for levels. Verify if student can access pathway and enter code review challenge.
- Teacher ends a code review challenge. Verify if teacher can collect the scores of the students.

## 6. Maintenance Plan and Details

A consistent maintenance plan is needed for the continuity of the product and to detect and solve possible problems at earlier stages.

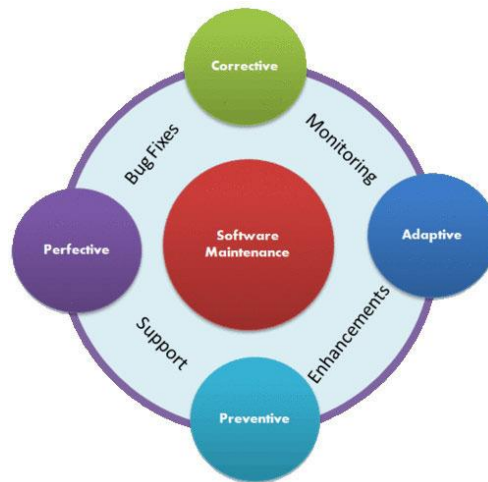


Figure 15: Maintenance Plan Approach

Maintenance of the Code Review Game is based on monitoring software actions and the data gathered from software activities. Maintenance plan provides highly adaptive software which bugs are quickly noticed and fixed and each improvement brings better user experience and approach to perfect software. IEEE's software maintenance standards are followed for sustaining accomplished maintenance plan [7].



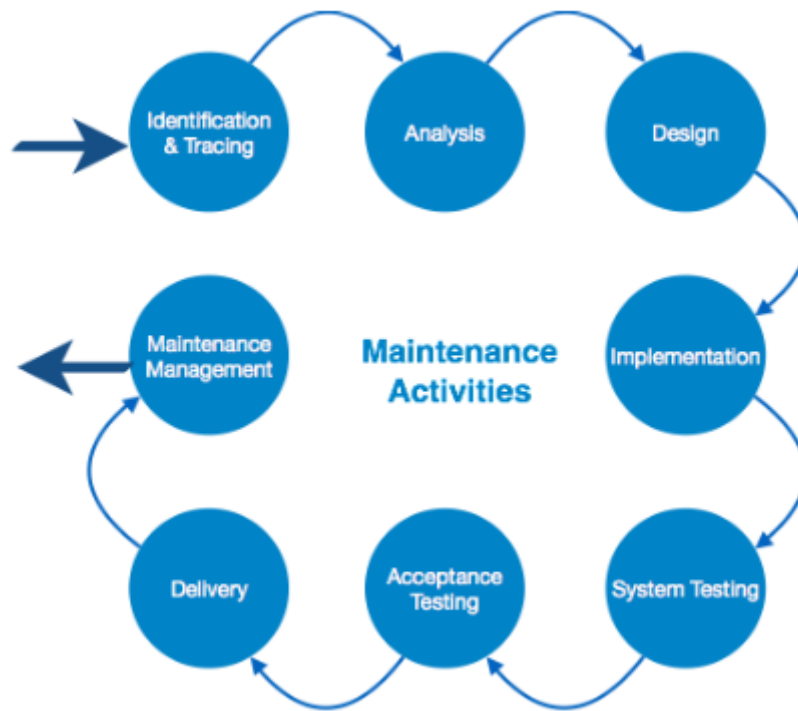


Figure 16: IEEE Maintenance Process Framework

First step of the maintenance plan is identifying and tracing the problem according to user feedback or system activities. Analysis is the phase which impacts of the problems understood. Solutions will be designed in the design phase by considering the integrations of technologies used in the implementation. Implementation will be completed according to design of the solutions and all tests related to changes will be completed. The changes will be deployed after successful tests and smooth integration.

## 7. Other Project Elements

### 7.1. Consideration of Various Factors

The impact of software products increases on a daily routine. Day by day people are seeing technology more intensely in their daily life thorough software products. As a result of this, some external factors like public health, public welfare or social factors start to affect the design and development of software products. In this case, the responsibility of the software designers and developers increase according to the share of technology in people's life.

The Game – Code Review is a product which aims to adopt code review habit for the students who learn software development. The target audience of the product is very limited so that it does not have a comprehensive effect for public health.

Everyday billions of people enter personal data to software applications and the developers of the products are responsible for protecting the private data from third parties. In our case, we have designed the code review game by considering the importance of privacy. E-Mail addresses, passwords or any other personal is not shared with any different application. On the other hand, code review is an important phase for revealing bugs in the written code before a release and some of the bugs can cause serious security problems for the software products. By the impact of our product, code review will be adopted by the freshman developers and this will increase the rate of finding security bugs in a program before a release. In one sense, secure software product means secure public.

The Game will give professional culture to freshman developers in terms of code review aspects. Usually, code review is not seen as a regular phase in the development process and it is not a settled culture especially for the less experienced developers. To provide a code review as a regular phase of development will change the professional culture of these developers.

The application gives chance for users to compete as a whole class. Reviewing a code script with classmates in a competitive way would increase the sense of togetherness with adopting a code review habits. The ability to work in a team is a very important earning for a developer because most of the software projects are developed by teamwork. So, our design will have positive effects on social factors in terms of interoperability and compete with respect. On the other hand, our game carries educational purposes. Education is totally related to social factors and our analysis is done by considering the educational purposes. We aimed to teach code review to freshman developers with all aspects in an enjoyable way by keeping the balance between education and entertainment.

The Game – Code Review has a very specific target audience and this situation isolates our solutions from the global, environmental and economic factors. Already, The Game is for the usage of freshman students and general concepts like global, environmental and economic factors are not deeply related to our target audience and product design.

	<b>Effect level</b>	<b>Effect</b>
<b>Public health</b>	0	Not a concrete affect
<b>Public safety</b>	5	Code review will reduce bugs and more secure products will be released with fewer bugs. It is important for data privacy.
<b>Public welfare</b>	0	Not a concrete affect
<b>Global factors</b>	0	Not a concrete affect because of the target audience
<b>Cultural factors</b>	4	Provide professional culture in software developments manners. Code review will be adopted as a regular phase of a development process.
<b>Social factors</b>	6	Improving teamwork and educate freshmen for the code review

Table 1: Factors that can affect analysis and design.

## 7.2. Ethics and Professional Responsibilities

The ethical and professional responsibilities of software engineers increase as the share of technology in daily life increases. This situation brings the necessity of making analysis in ethical manners with professional approaches. To meet these requirements, data privacy, information security, and copyright issues are prioritized in our product.

There is a risk of a teacher to upload a copyrighted piece of code to the program and this case will cause an ethical issue. So before any sort of upload to the program, the uploader should agree on a text that he/she takes the responsibility of the uploaded code.

The program is storing store personal data of users and it contains the game results of contributors which will assign their rankings. As the developers of the program, we won't share any sort of personal data with third parties and we will not allow sharing any sort of data sharing with third parties without the permission of the particular contributor. Also, we won't use any sort of data for our own purposes.

At the beginning of the program we clearly state all ethical rules to users for avoiding possible professional and ethical issues. In terms of profession, we worked for producing high standard software with qualified features in it.

The impact of our engineering is mostly related to the societal context because our product carries educational purposes. In this manner, we are responsible for creating a useful and functional learning environment and our decisions is taken regarding our purposes.

### 7.3. Judgments and Impacts to Various Contexts

<b>Judgment Description</b>	Data privacy is important and protected by law.	
	<b>Impact Level</b>	<b>Impact Description</b>
<b>Impact in Global Context</b>	6	Personal data collected nationally are used at a global level. However, the use of personal data is mandatory to obtain permission from the user, and each country has its own legal limits.
<b>Impact in Economic Context</b>	3	Share of personal data with anonymous third parties may result in access to bank accounts and result in economic losses.
<b>Impact in Environmental Context</b>	0	No environmental context.
<b>Impact in Societal Context</b>	10	With increasing awareness on data privacy, societies have started to be sensitive about this issue. People's expectation is that software has to be sensitive to the use of personal data and data is not shared with unknown third parties.

Table 2: Judgment Description 1

<b>Judgment Description</b>	Code review is an important process for software development process. It should be taken seriously and not to be skipped.	
	<b>Impact Level</b>	<b>Impact Description</b>
<b>Impact in Global Context</b>	3	In the scope of global software projects, code review became a fundamental step in project development process with the raise of agile methodologies.
<b>Impact in Economic Context</b>	2	A good code review habit shortens the project development process, resulting in reduced costs.
<b>Impact in Environmental Context</b>	0	No environmental context.
<b>Impact in Societal Context</b>	5	A good code review helps developers to produce clean, useful software. If we take into account that human life is dependent on software products, useful software would have a positive effect on societies.

Table 3: Judgment Description 2

<b>Judgment Description</b>	Although an important step, code review may not be taken seriously or it can be skipped. In order to understand that the code review phase is a basic process in developing the project, it is necessary to gain a code review habit from the early times.	
	<b>Impact Level</b>	<b>Impact Description</b>
<b>Impact in Global Context</b>	2	Today's CS students will become the developers of tomorrow. Developers who have code review habits would help to produce cleaner software.
<b>Impact in Economic Context</b>	0	No economic context.
<b>Impact in Environmental Context</b>	0	No environmental context.
<b>Impact in Societal Context</b>	5	Software products developed with the code review phase would offer a better user experience.

Table 4: Judgment Description 3

## **7.4. Teamwork and Peer Contribution**

Every member of the team tried to give his best for each of the development process. Because of everyone's different backgrounds and expertise, of course we could not work equally at every stage, but everyone showed maximum devotion in the subjects he knew. Everybody tried to help each other at every missing point. Specifically, if we come to the fields where everyone works, Burak was the technical advisor of the group. He established the necessary infrastructure by using the appropriate technologies for the project. Also he hugely worked on frontend and backend development. Okan was responsible from frontend development and also contributed documentation by determining the requirements of the projects. Mert contributed to development of project plan and frontend development. Berk was responsible from documentation and tracking the project status and deadlines for meeting project objectives. Also he partially contributed on both frontend and backend development. Throughout the year, all members sustained the collaborative team environment and took the lead when necessary for documentation and development even there was pandemic conditions.

## **7.5. Project Plan Observed and Objectives Met**

The objective of the project is to develop an application that is turning code inspection into more effective and amusing process. This application will be used by instructors and students and students will be able to solve exercises posted by teachers through the application at the end.

### **Important milestones of the project:**

- Project Specifications
- Analysis Report
- High-Level Design Report
- Low-Level Design Report
- Final Report
- Presentations & Demonstrations

**Objectives defined:**

- Design user stories
- Design review scripts
- Gamification Design Framework
- Define Solution Concepts
- Achieve Application Adaptation
- Achieve Implementation

We have determined the documents we need to deliver and the delivery of the project as our milestones. In this context, we successfully delivered our reports and completed the implementation of project. Each milestone had different objectives. Designing user stories and code review scripts were important for analysis reports. Gamification design framework and defining solution concepts were objectives for high-level and low-level design reports. Achieving application adaptation and completing implementation were important for the implementation and development phase of the project. In terms of meeting objectives, designing user stories, designing review scripts, having gamification design framework and defining solution concepts are done successfully. Also application adaptation and implementation achieved by implementing the planned requirements.

**7.6. New Knowledge Acquired and Learning Strategies Used**

For the project, each group member aimed to improve himself in technological manner and documentation. Background of each other is totally different but we tried to contribute equally in each phase of the project. To achieve this, we chose trend technologies which can be useful for us in future careers. We implemented the frontend in AngularJS which is a popular JavaScript framework. AngularJS is totally new for some of us and we are going to learn AngularJS together. We implemented the backend by Django framework. Already we all know Python but Django framework was new to some of us. Those members covered Django by themselves and asked for help to other members when needed.

We mostly used internet sources to learn all of these technologies. Already official documentations are published for each of them and there were dozens of tutorials we could follow. The other important option for learning was asking each other. We stated our understandings and missing regularly to each other for checking progress for learning. We directly helped each other when needed.

### **List of Learned Technologies:**

- Django
- MongoDB
- Karma Test Runner
- AngularJS
- Django Test Engine
- Selenium

## **8. Conclusion and Future Work**

When the project started, a game intended for education was planned by creating a competitive environment. Looking at the end of the project, a game has been created that provides the necessary conditions which are consistent with the plans. With the agile methodology becoming popular, the code review phase is now one of the key stages in software project development. Despite this situation, sometimes code review phase is not taken seriously or can be skipped. This game was designed with the fact that code review habit should be gained at the beginning CS education by presenting it as an important step for software development. As a result the application is suitable for this purpose by the objectives met.

Most of the technology companies in the information technologies are using their own frameworks and they all have defined naming conventions and coding rules for the development process of software. In terms of future work, The Game can be customized according to needs of companies. Companies can use this software for their orientation programs to help new hires for grasping the company's coding culture.

## **9. Glossary**

- **Code Review Challenge:** The main point of the game. Gaining scores by finding defects on a code script.
- **Pathway:** Group of code review challenges which are ordered in difficulty levels.



## 10. References

- [1] Otto, M., & Thornton, J. (n.d.). Bootstrap. Retrieved from <https://getbootstrap.com/>
- [2] What Is MongoDB? (n.d.). Retrieved from <https://www.mongodb.com/what-is-mongodb>
- [3] Christie, T. (n.d.). Django REST Framework. Retrieved from <https://www.django-rest-framework.org/>
- [4] AngularJS. (2020, April 4). Retrieved from <https://en.wikipedia.org/wiki/AngularJS>
- [5] (n.d.). Retrieved from <https://startbootstrap.com/template-overviews/sb-admin-2/>
- [6] (n.d.). Retrieved from <https://angular.io/guide/testing>
- [7] IEEE Standard for Software Maintenance," in IEEE Std 1219-1998 , vol., no., pp.1-56, 21 Oct. 1998, doi: 10.1109/IEEESTD.1998.88278.