HOUSE OF WORDS WEB APPLICATION

Project report submitted in partial fulfillment of the requirement for the degree of Bachelor of technology in

Computer Science and Engineering

By

Deepanshu Sharma (151286)

Mohit Sood (151288)

Manmayur Kaur(151292)

Manika Sharma (151294)

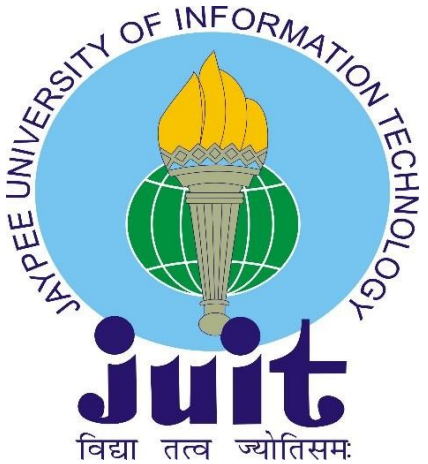
Sakshi Sharma (151297)

Kartik Sharma (151299)

Rohan Arora (151326)

Under the supervision of Dr Hemraj Saini,

Department of Computer Science & Engineering and Information Technology



Jaypee University of Information Technology Waknaghat, Solan-173234, Himachal Pradesh

Candidate's Declaration

I hereby declare that the work presented in the report entitled "**House Of Words Web Application**" in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science and Engineering/Information Technology** submitted in the department of Computer Science & Engineering and Information Technology, Jaypee University of Information Technology Waknaghat is an authentic record of my own work carried out over a period from 5 April 2019 to 17 May 2019 under the supervision of **Dr Hemraj Saini.**

The matter integrated in the report has not been submitted for the award of any other degree/diploma.

Deepanshu Sharma (151286)

Mohit Sood (151288)

Manmayur Kaur(151292)

Manika Sharma (151294)

Sakshi Sharma (151297)

Kartik Sharma (151299)

Rohan Arora (151326)

This is to certify that the above made statement by the candidates is true to the best of my knowledge.

Mr. Munish Sharma,

Member – Education, Training & Assessment, Infosys Dated:17/05/2019

ACKNOWLEDGEMENT

We would like to express our special thanks and our intense gratitude to our Project Mentor **Mr. Munish Sharma** who helped us in all the phases of the project development of **"House Of Words Web Application"**. Under his guidance we were able to complete all the functionalities of our project successfully. We would also like to expand our gratitude to all those who have directly/indirectly helped us in completion of our project. The opportunity to do this wonderful project helped us in doing a lot of research work and learn various technologies. We owe our debt to our Project Mentor **Mr. Munish Sharma**. We would also like to thank our respected faculties at Jaypee University of Information Technology who made us this capable so that we were able to complete our tasks efficiently and on time.

TABLE OF CONTENTS

LIST OF FIGURES

20	Select Genre Page
21	Help Page
22	Login Page
23	Leaderboard Page
24	Game Time up Page
25	User profile page
26	User Home Page
27	Home Page
28	Game Stats Modal
29	Reset Password Page
30	Reset Password Page
31	Feedbacks
32	Add Advertiser

LIST OF TABLES

LIST OF GRAPHS

LIST OF ABBREVIATIONS

S. No.	Abbreviation	Definition
1	MVC	Model View Controller
2	DB	Database
3	UI	User Interface
4	ER	Entity Relationship
5	SDLC	Software Development Life Cycle
6	IIS	Internet Information Services
7	EF	Entity Framework
8	JS	JavaScript
9	ASP	Active Server Page
10	ES6	ECMAScript 6
11	SQL	Structured Query Language
12	CRUD	Create, Read, Update, Delete
13	SPA	Single Page Application
14	API	Application Program Interface
15	JSON	Java Script Object Notation
16	XML	Extensible Markup Language
17	REST	Representational State Transfer
18	URI	Uniform Resource Identifier
19	JWT	JSON Web Tokens

ABSTRACT

In our project House of Words web application, we worked on an online word puzzle game which utilizes a 4X4 matrix of jumbled alphabets. It is a single player game. It also consists of various genres which an individual can choose on their own. We divided our application into two parts: frontend (Client facing application) and backend (Server Side).

We leveraged modern web technologies like Vue Js for building user interface, bootstrap for styling and ES6 for handling user events and dynamic behavior of the application. We also used various third party libraries for better development experience and for reduction of development time. For backend, we utilized ASP.net MVC, EF and JWT tokens for authenticating user. For database we have used Microsoft sql server.

We performed various CRUD operations for advertisements, feedback, user details, etc. Our frontend application is a SPA which provides a good user experience.

Since it is an entertainment we tried to make our application as aesthetically pleasing as possible. This game also supports multi user access. Users can play the game at the same time from different locations without any problem.

CHAPTER 1 INTRODUCTION

1.1 Introduction

1.1.1 MVC Architecture and Web API

The MVC architecture is a design pattern which was built to make web application more modular, robust and manageable. The main components of this model are Model, View and Controller. The first component that is the model is has the core data and functionalities. The second component that is view is used to display the information on frontend for the end users. The last component that is controllers are to manage the user input and proper functioning between frontend and backend. Controller is also responsible for rendering the view of the application. It splits an interactive application into three departments that is i/p, o/p and processing.

View components are visible to users. A user can make a request for data or service from the view itself and this request is processed by controller part of MVC. API refers to application programming interface. API is used for communication between two or more applications irrespective of technologies with which applications are built. Web API refers to an API running over internet. Web API mostly utilizes JSON or XML data for communication. A web API can be used by any software or platform from anywhere and it doesn't require any special software for integration. Only an internet connection is sufficient. A Web API can be accessed from browsers, terminal etc. It can also be consumed by any frontend frameworks easily. It is common to use web API's to build RESTful web services.

REST is a design pattern in which the server doesn't have any knowledge of the state of an application. In RESTful architectural design everything is treated as a resource which is uniquely identified by a URI which is generally a unique id stored in database.

1.2 Problem Statement

'House of Words Web App' is an online web portal that provides a platform to play online word puzzle game. The game will offer puzzles related to English language and it consists of a 4*4 matrix with jumbled alphabets. The game admin will have its business running with the number of players at any point of time and also with the promotional advertisements on the web app. The game application should allow sign in or play as a guest option. The guest will have access to limited functionalities. A logged in user will have access to all the functionalities. A timer should be maintained on the game top and the game should run for maximum of 2 min. The game should have a dedicated space for commercial advertisements. The game should also maintain a DB of advertisements from different advertisers. The advertisers should be billed based on the time the advertisements are displayed. Users should also be able to see their ranking on the leaderboard. Only logged in users should be able to submit the feedback for the game.

1.3 Aim and Objectives

The aim is to create a web-based online game that offers a puzzle related to English language words.

1.4 Methodology

In this project, we would use ASP.net core MVC and EF for implementing the backend. For frontend we would be using Vue Js (A JavaScript framework for building user interface). We would also be using a third layer which would be acting as an intermediate between backend and frontend. All the requests coming from client side will first go through this web API and then to DAL which will then access the data from database and return it to web API. API will then send the response to Client side in JSON format. This JSON data then can be accessed by Vue Js application to display appropriate things to user. API calls can be made from front end application using a third party module called Axios. JWT tokens are used for user and admin authentication. This token is generated if

user/admin login is successful. The token is stored in the browser's local storage. This token
is then sent with every request to backend where it is decoded and verified for user authentication. JWT tokens can also be given an expiry time. This token will be cleared from browser's local storage once the user logs out.

AGILE methodology development is a development that involves continuous iterative software development. It includes testing and maintenance of the software as well as the improvisations in the SDLC of the project. In Agile software development requirements and solutions evolves through collaborating the self-organizing cross-functional teams. It promotes a disciplined project management process which encourages inspection and adaption. It encourages team work, self-organization and accountability. It includes a scrum master who conducts daily scrum meetings manages the agile sheet. In scrum meetings all team members should give a brief about the work they have to do and also the pending work. The agile sheet includes user stories , product backlog, Dashboard, Reference data, capacity planning, sprint backlog and standup meetings.

1.5 Organization of Project Report

Chapter1, covers the basic introduction about the project i.e. what technologies are used. Functionalities of project, the approach and the methodologies used.

Chapter2, covers the literary review which we have gone through

Chapter 3, includes various project requirements, DFD's, ER diagram, user case diagrams and various architecture diagrams.

Chapter 4, It includes the test plan and the performance analysis of the project.

Chapter 5, concludes the project and also gives a brief about future scope.

CHAPTER 2 LITERATURE SURVEY

Literature survey is a method used to evaluate and understand the researches and high quality work done in a particular area. The primary focus is to derive a calculative structure and evaluation of the topic through various understandable techniques. The main objective is to understand the core structure of MVC architecture, VUE and its various methodologies.

2.1 Understanding the MVC architecture effectiveness to create web applications.

MVC stand for MODEL-VIEW-CONTROLLER. It is one of the most widely used web development framework which is used to create a variety of projects in the market.
The MVC architecture consists of three logical components:

- The Mode
- The View
- The Controller

Each of these logical component is responsible for a particular aspect in an application.

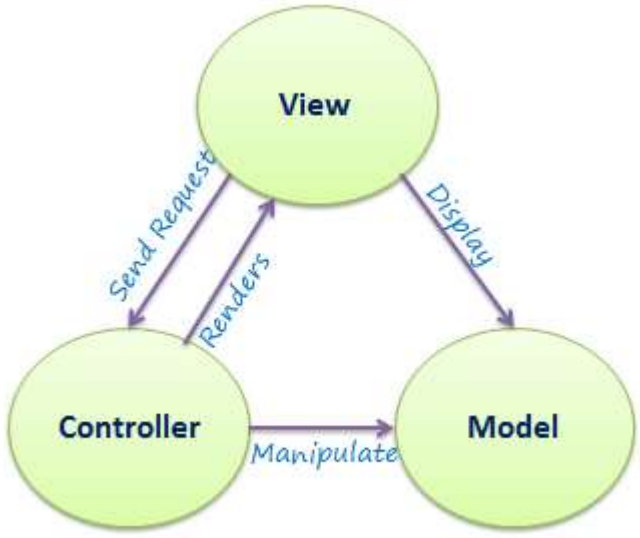


Fig *i.* Model-View-Controller Architecture

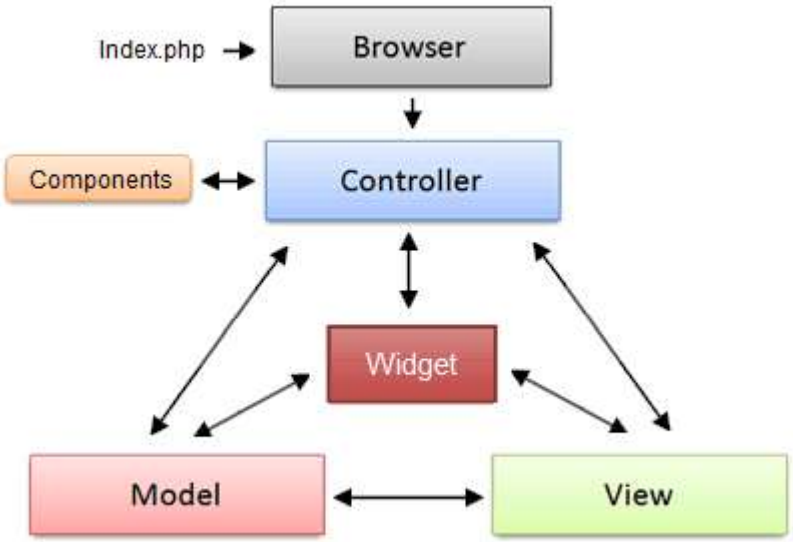


Fig *ii*. Model-View-Controller Work Flow

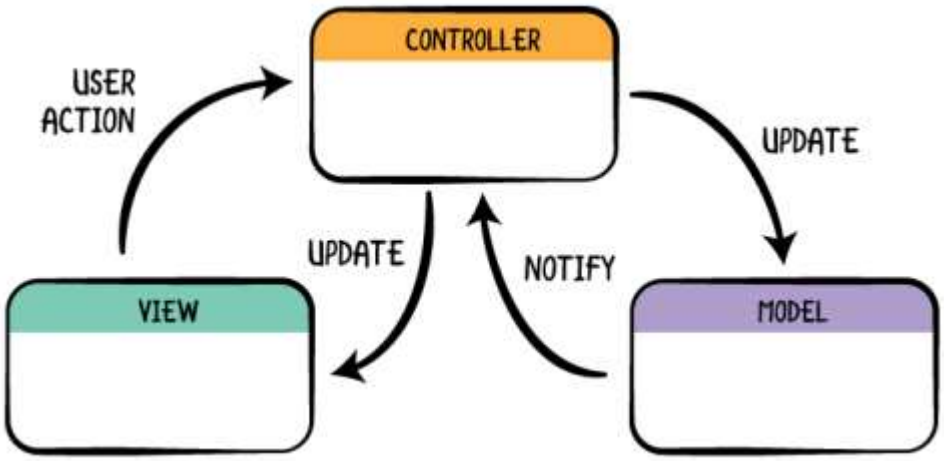


Fig *iii*. Interaction between MVC Components

2.2 Understanding the MVC components workflow

Each component has specific functionalities.

- Model: Responsible for logically related data in an application.
- View: Creates a real implemented view of the model.
- Controller: Determines the impact of the inputs on the actual interface.

We can fictionally study these components as a movie story by relating model as the actual script. The changes done in the can be interpreted as the components and the View can be treated as the actual story clip.

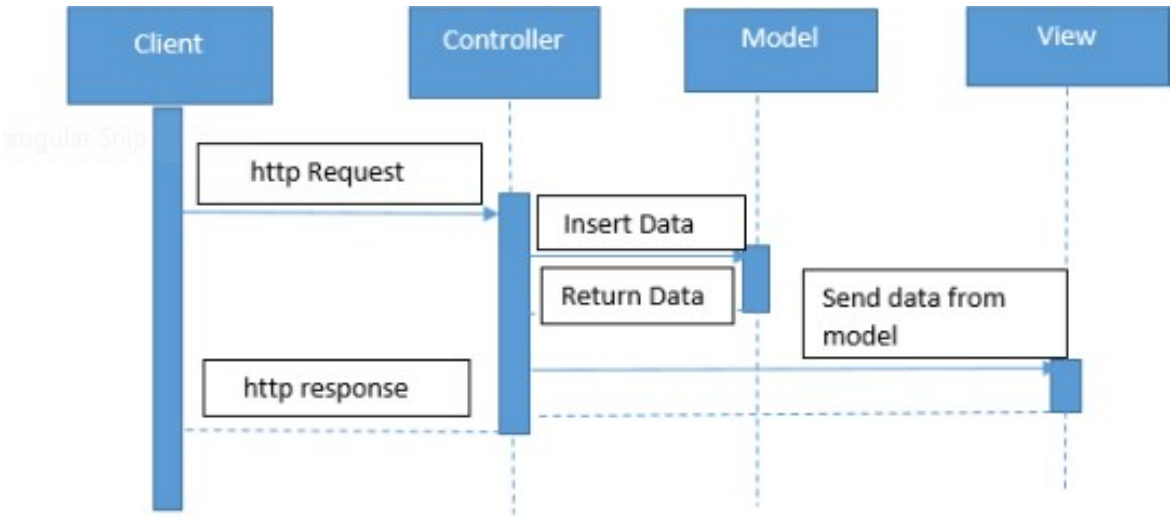


Fig *iv*. MVC WORK FLOW

2.3 Understanding Vue Js

One of the widely used framework that has astonished the market with its easy to implement functionalities. A JavaScript Framework that provides a variety of optional tools for building user interfaces.

Fig *v.* VUE.JS

2.3.1 Advantages of Vue Js

- Small Size: Basically the success of a JavaScript Framework depends completely on its size. The size of this framework is small consisting of only 18-25KB and has a very less downloading time.
- Easy to understand and implement: It has a simple structure which is easy to understand and the user can quickly use it in his project to enhance his functionalities.
- Detailed and Precise Documentation: The developers usually prefer a detailed documentation while developing their project which helps them to understand the functionalities and implement them with proper understanding.
- Flexible by nature: It is flexible in nature. The user can use HTML, CSS, JavaScript for writing the templates. It also makes easy for the developers to understand.
- Two Way Communication: It enables two-way communication because of its flexible structure.

CHAPTER 3 SYSTEM DEVELOPMENT

3.1 HW/SW requirements

System Detail	Specification
Processor	Intel Dual core
RAM	2GB
Operating System	Windows7
HDD	160GB
Speed	2 GHZ and more
Language	C#, CSHTML and JavaScript
IDE	Microsoft Visual Studio 2017,Visual Studio Code
DBMS	Microsoft SQL server

Table 1. HW/SW requirement

3.2 Requirement Specifications

"House of words web application" is a web portal for playing games i.e. online word puzzle. This application is built using ASP.net, VUE Js, Bootstrap, EF for frontend and backend. This game includes user friendly interface. It is easy for a user to learn the rules of the game using a help button. A third layer is also used that will behave as an intermediate between the front end and backend. The requests generated on client side

will pass through the API and then DAL to gather data from database and pass it back to web API. AXIOS is a module which is responsible for making API calls to the frontend. For authentication we are using tokens that is JWT tokens. Only if the user /admin login is successful i.e. authenticated then the token would be generated. The tokens will also have an expiry time for security reasons. Once the token is expired it will be deleted from the local browsers location. The login is only successful when the token is authorized.

3.2.1 Functional Requirements

A user should be able to register. A user should be able to login to the web application provided he enters a correct combination username and password. If login is successful, he/she should be able to access the following features:

1. Users can view their profile details.
2. Users can also update their profile details.
3. Users can also reset the password if forgotten.
4. They can provide the feedback and can also update it.
5. They can also play the game any number of times.
6. They can also restart the level anytime they want and can also skip the question(s) if stuck.
7. Users can also exit the game anytime they want.
8. Users can also view their rank and score from their profile and can also have look at other users ranking.

Guest user:

1. Can play the game with limited features.
2. Can view leaderboard but his score and rank won't be calculated.
3. Cannot give feedback for the game.
4. Can view his stats after the game.

Admin will have access to the following features once he is authorized:

1. Can view all user feedbacks.
2. Can manage the advertisers and advertisements.
3. Can ban the users.
4. Can perform CRUD operations on advertisements, advertisers.
5. Can also delete abusive feedbacks.

3.2.2 Non-Functional Requirements

- **Security:** System should have sufficient security measures in place to restrict unauthorized user access.
- **Error logging:** Appropriate user messages should be displayed in case of any launch failure, server error or any other error. Maintaining logs for whole system is not mandatory.
- **Performance:** Queries should be written in such a way to reduce the latency involved in processing and fetching data from database.
- **Scalability:** Multiple user should be able to access the web-app.
- **Availability/Reliability:** Downtime of the application should be very less.

3.2.3 Entity Framework Approach

Entity framework is a data access framework used for creating and testing data in visual studio. We used this in our asp.net MVC app.

This includes various approaches i.e.:

1. Code first approach: This approach allows us to define our model using C#. In this approach the database will be created after the code is done.
2. Database first approach: In this approach the database is provided first and then the model codes are created.
3. Model first approach: In this the entities, relations and inheritance hierarchies are

designed and then the database is generated from the model built so far.

We have built our project using the DB first approach as all of the team members were familiar with it and learning code first approach would have been lapses in the project time line.

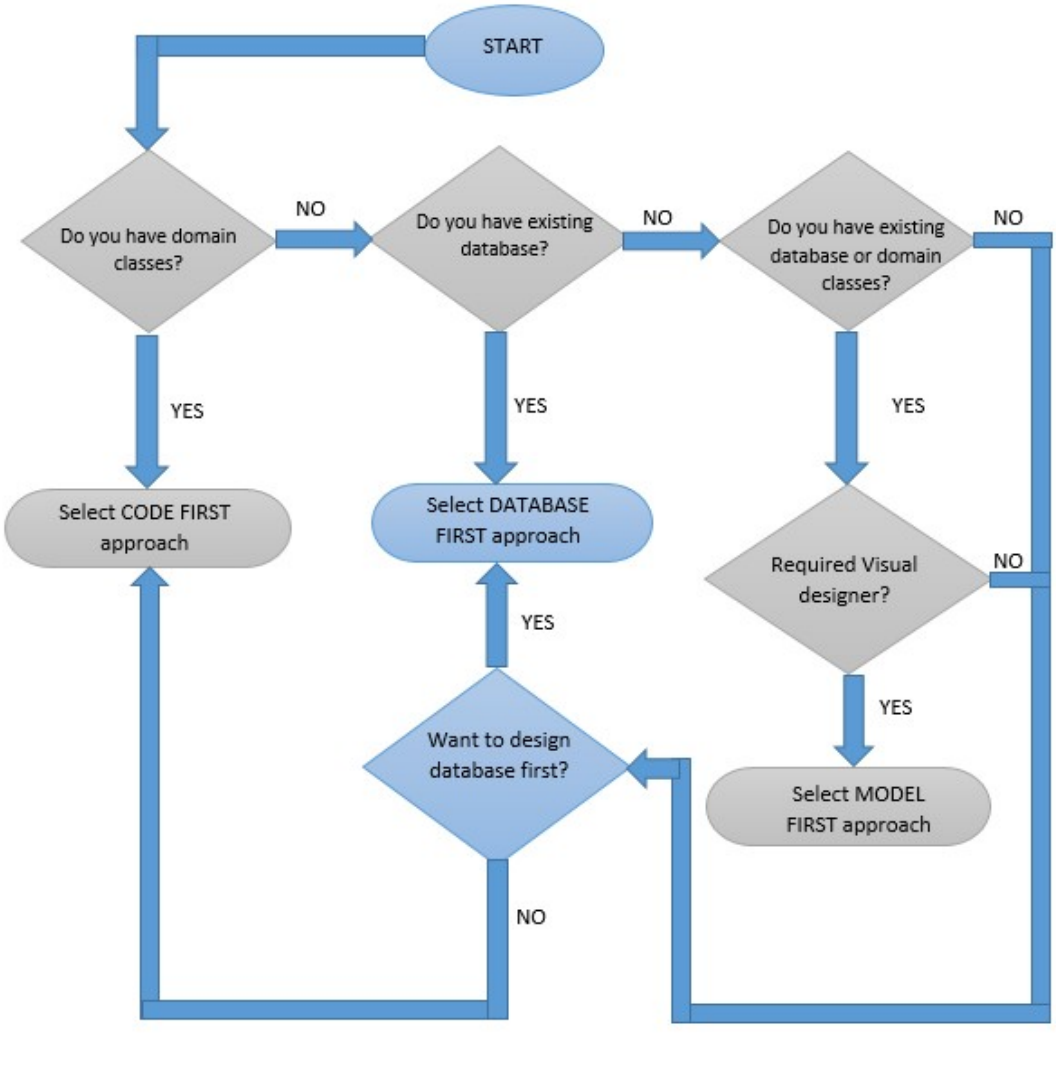


Fig. *vi* Approaches to A Project

3.3 Data Flow Diagram:

The flow and the working of the project is explained with the help of the Data Flow Diagrams(DFD). The detailed working of the components can be explained with the help of DFD standards.

There exist variety of DFDs that varies according to the structure of the process. The most frequently used DFDs are mentioned below.

Why DFDs are the best choice?

- Distinguishing the processes and dividing them into modules.
- Explaining the flow and the structure of the processes.
- Provide a correct sequence to the flow of activities.
- Arranging the tasks of the modules in the flow.
- The pointing in and out arrows helps to explain the structure of the processes much better.

3.3.1 DFD: User Login

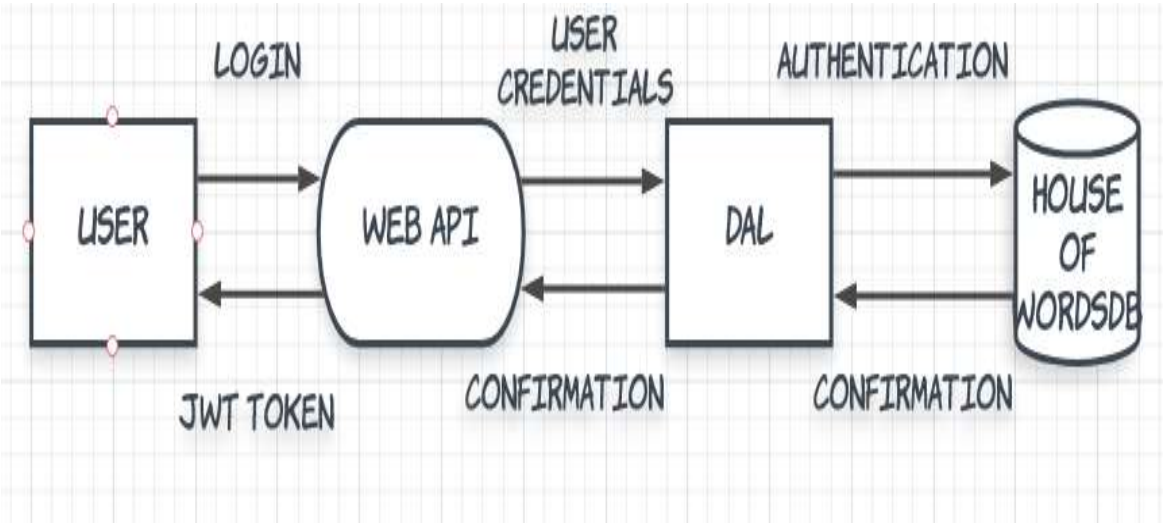


Fig *vi.* Data flow diagram of user login

3.3.2 DFD: User Registration

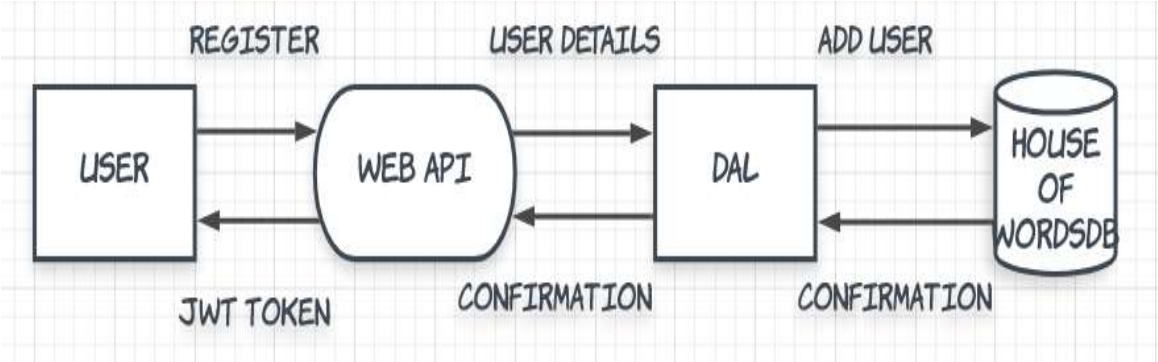


Fig *vii.* Data flow diagram of user registration

3.3.3 DFD: Admin Login

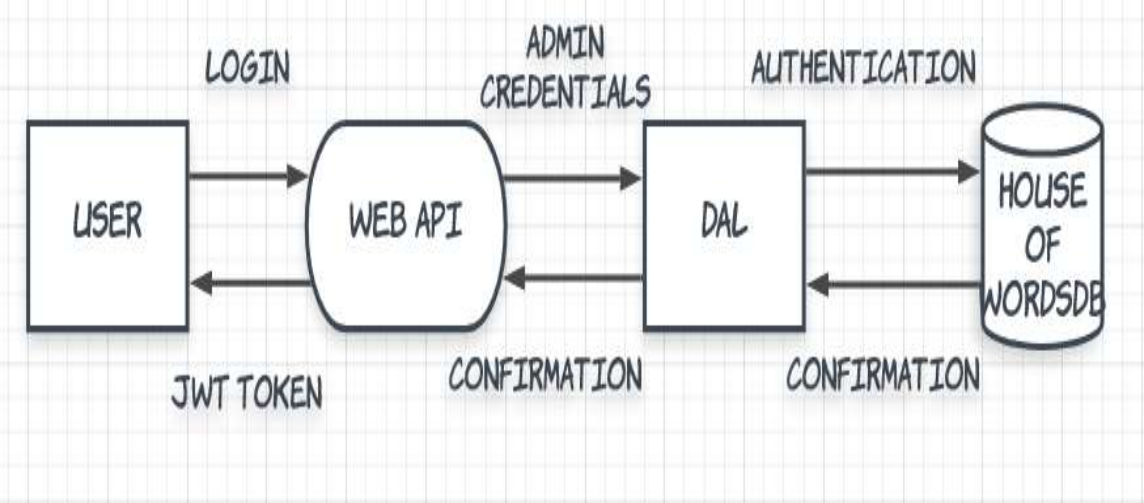


Fig *viii.* Data flow diagram of admin login

3.3.4 DFD: Guest User

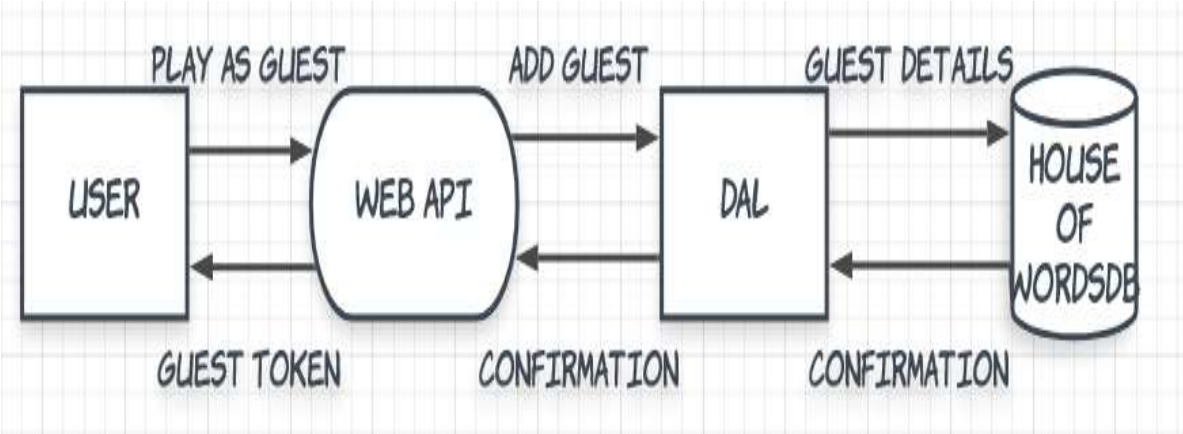


Fig *ix.* Data flow diagram of guest user.

3.3.5 DFD: Reset Password

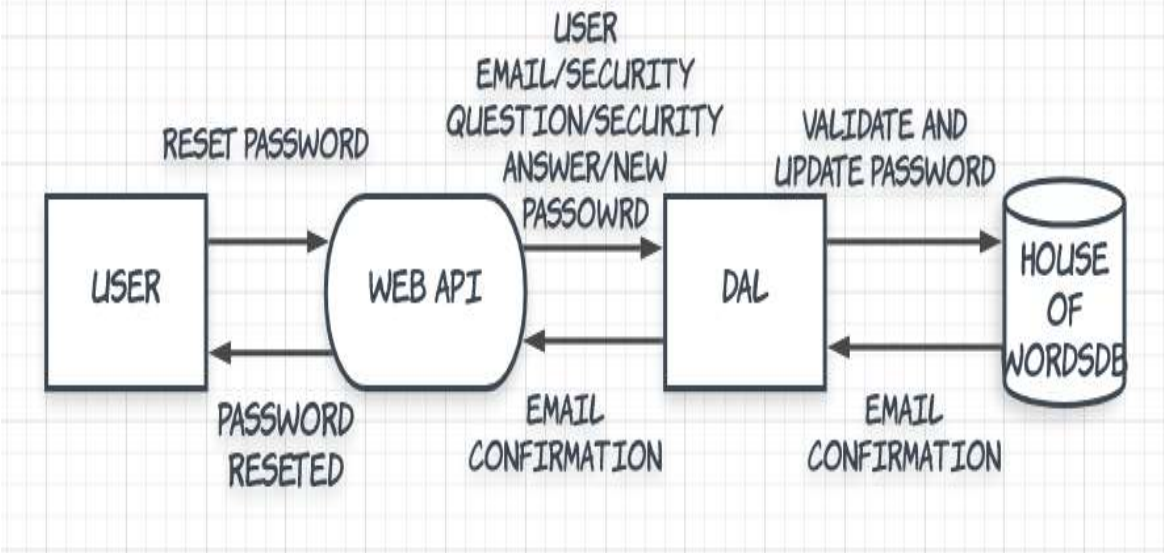


Fig *x.* Data flow diagram of reset password

3.4 ER Diagram:

Entity relationship diagram is a model which is used to define a relationship between the various entities in the field of interest. The below diagram displays the relationship between the various entities stored in the database.

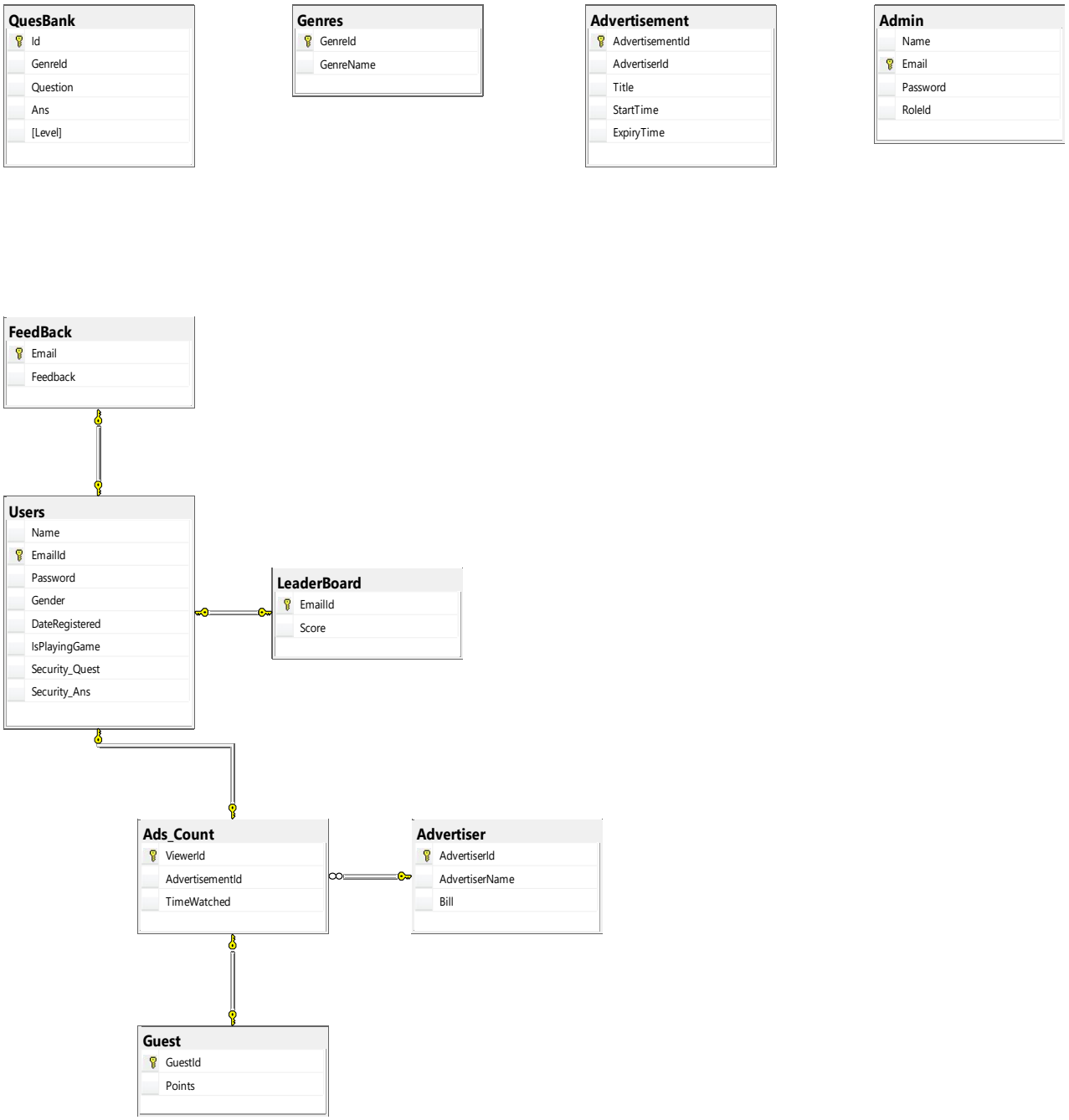


Fig *xi.* Entity relationship diagram

3.5 Use Case Diagram

1) User

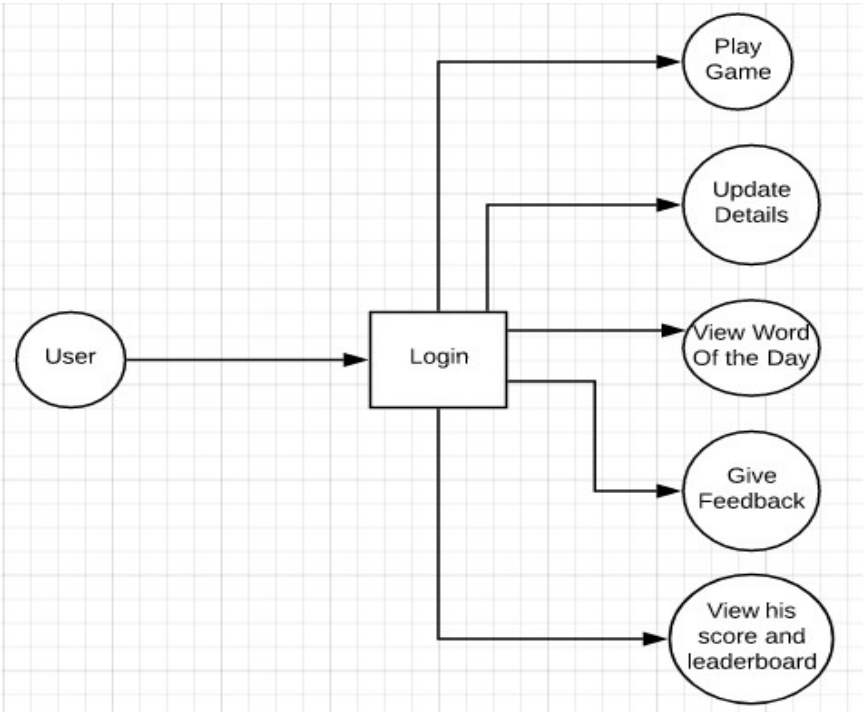


Fig *xii.* Use case diagram for user

2) Admin

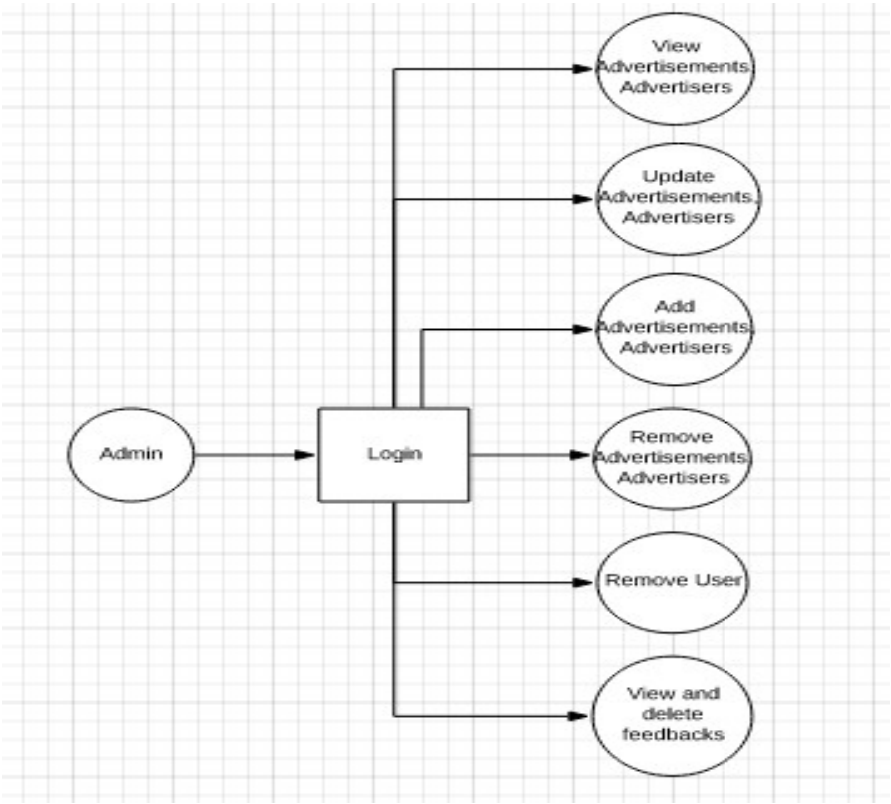


Fig *xiii.* Use case diagram for admin

3) Guest

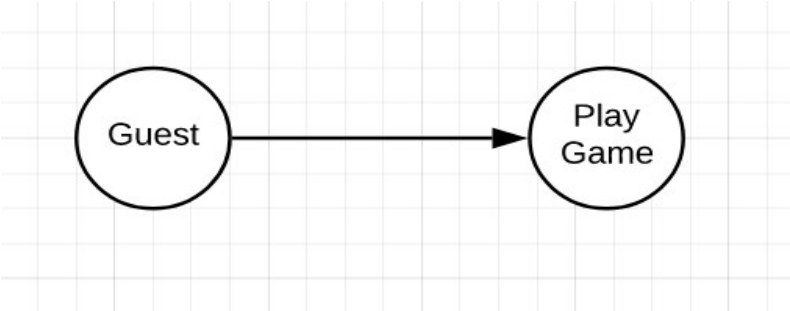


Fig *xiv.* Use case diagram for guest

3.6 High Level Design Architecture

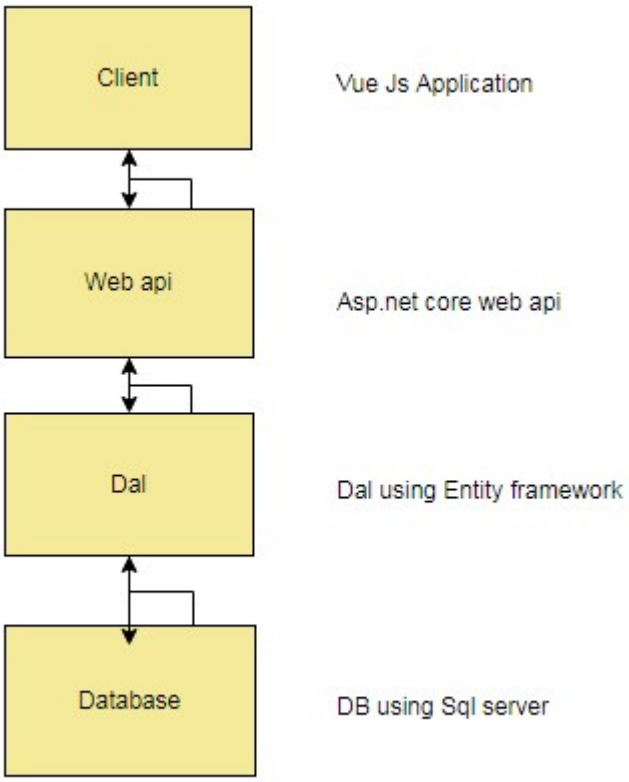


Fig *xv.* High level abstraction of system design

3.7 Database design

S. No	Table Name	Description	Field Name	
1	**ADMIN**	Details of Admin.	Name	VARCHAR(50)
			Email-ID	VARCHAR(50)
			Password	VARCHAR(20)
2	**USERS**	Details of users.	Name	VARCHAR(50)
			Email-ID	VARCHAR (50)
			Password	VARCHAR(20)
			Gender	CHAR
			Date Registered	DATETIME
			Security Question	VARCHAR(100)
			Security Answer	VARCHAR(50)
3	**FEEDBACK**	Feedback of the users along with their email id	Feedback	VARCHAR(50)
			Email-ID	VARCHAR(50)
4	**GUEST**	Details of the user playing as a guest .	Guest ID	VARCHAR(50)
			Points	BIGINT
5	**ADVERTISEMENT**	Advertisements to be displayed	Advertisement ID	VARCHAR(20)

			Advertiser ID	VARCHAR(20)
			Title	VARCHAR(100)
			Start Time	DATETIME
			Expiry Time	DATETIME
6	**ADVERTISER**	Details of the person giving the ads.	Advertiser ID	VARCHAR(20)
			Advertiser Name	VARCHAR(50)
			Bill	FLOAT
7.	**LEADERBOARD**	Details of the rank of the users playing the game.	Email-ID	VARCHAR(50)
			Score	BIGINT
8	**GENRES**	Various genres on which the questions in the game are divided.	Genre ID	VARCHAR(10)
			Genre Name	VARCHAR(20)
9	**QUES BANK**	Questions according to genres in the game	Genre ID	VARCHAR(10)
			Question	VARCHAR(200)
			Answer	VARCHAR(200)
			Level	TINYINT

Table 2. Database Design

CHAPTER 4 PERFORMANCE ANALYSIS

4.1 Agile Methodology

This methodology is a way that gives continuous iteration of developing and testing the entire software development life cycle. In waterfall model the testing and the development

is not a concurrent process but it is not so in case of agile methodology. In this methodology the testing and development goes side by side i.e. is a concurrent process. The advantage of this methodology is that the requirements can change at any point of time can be implemented efficiently. The requirements can change with the user demand and the software needs.

We have divided the project into different sprints. The sprints tell us how we have initially planned for the work flow of the project and how it actually is going. It gives us the overview of the performance of every individual. The analysis is done on basis of the graph which is formed at the end if a particular sprint

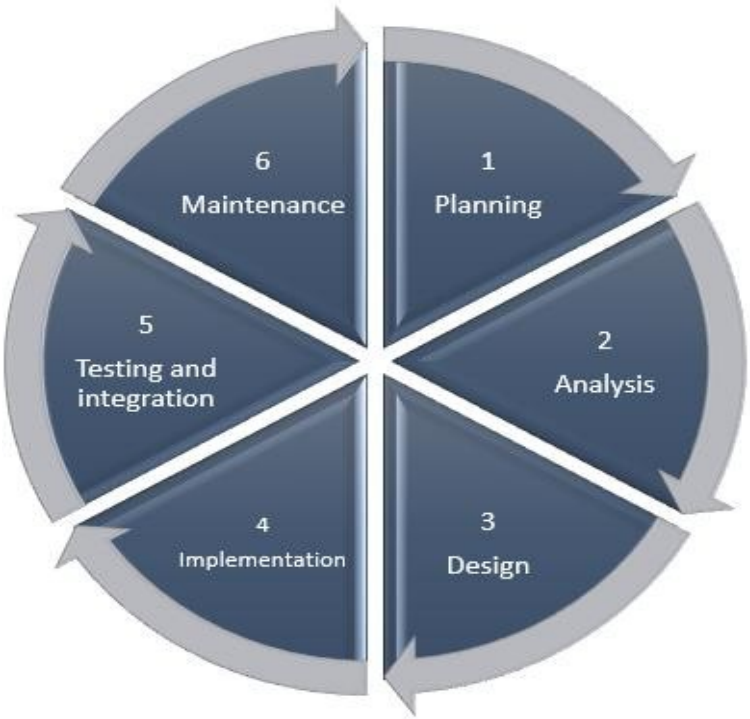
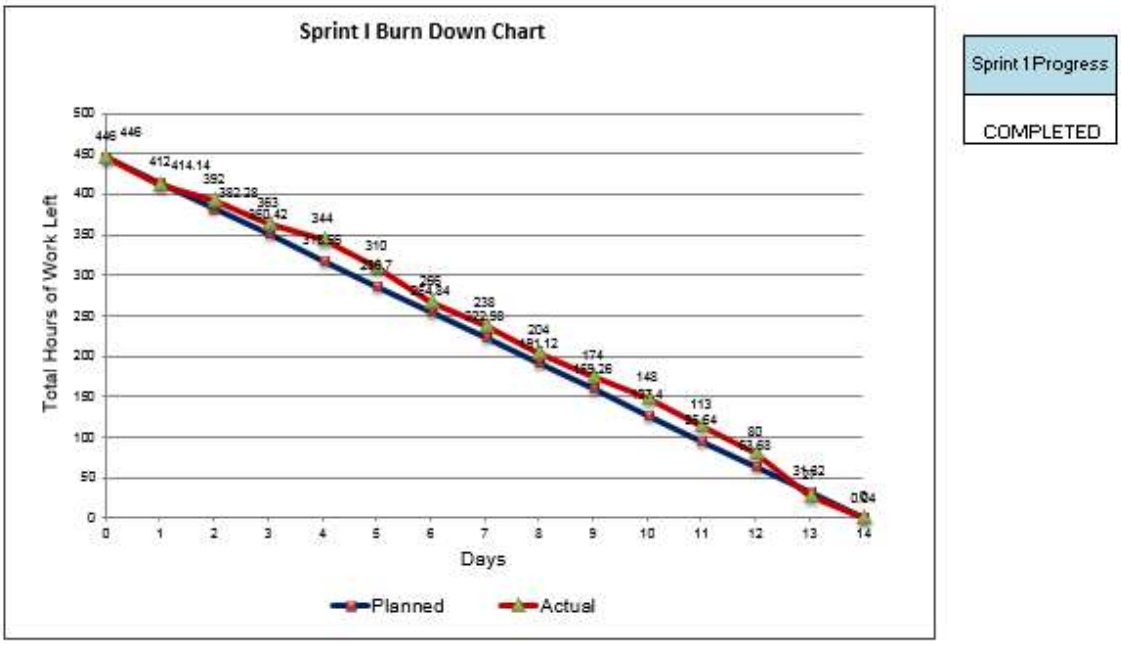
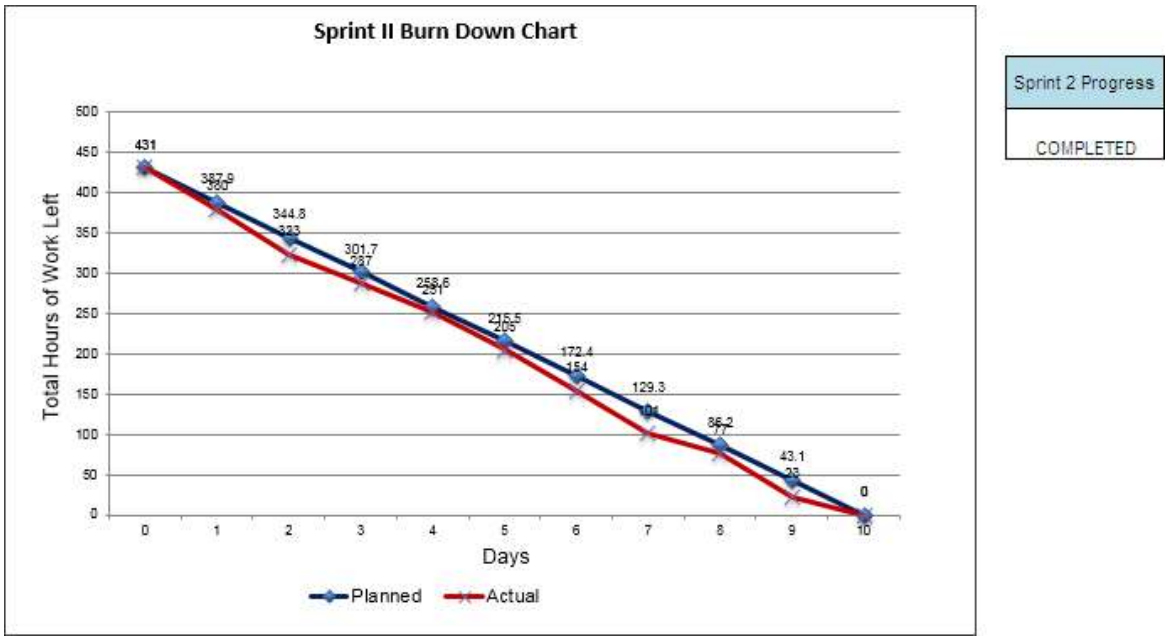


Fig *xvi.* SDLC



Graph 1. Burn Down Chart Sprint 1



Graph 2. Burn Down Chart Sprint 2

4.2 Optimization of algorithms, data access and performance

Algorithms have been optimized to increase efficiency and speed of frontend and backend application. Indexing have been used to increase access speed of data from data base as it uses hashing internally. Code files have been minified to reduce network latency introduced due to transfer of large data files. Use of Vue Js framework has improved the frontend application speed as it improves the document object model updating process which is otherwise very slow if we use vanilla JavaScript. Vue Js creates a virtual document object model and then compares it with the actual document object model and only updates the nodes which have changed. This decreases the load time drastically. Vue also helps in managing state of the application a piece of cake which is very difficult if we use vanilla JavaScript. DRY (don't repeat yourself) approach is used while building the project to make it clean, modular and less coupled as possible.

4.3 Screen field validations, defaults and attribute tables

Data annotations and frontend validation have been used in or project with proper meaning in each and every text field. Form unique identification we used the identity. Table constraints have been used for designing the portal.

4.4 Test Plan

Software Testing is a technique required to test the modules. The modules are tested at various phase and it is ensured that the output at various phases matches the expected or the planned outcomes. Though there are various methods used to test a software but we have preferred White Box Testing for the modules.

4.4.1 Login Functionality Testing (User)

Test case no.	Case	Expected Outcome	Actual Outcome
TC1	Email:abc@gmail.com Password: 1234567	**Error**: Invalid Credentials. Login failed.	**Error**: Invalid Credentials. Login failed.
TC2	Email: a@gmail.com Password:	**Error**: Password required.	**Error**: Password required.
TC3	Email: a@gmail.com Password: 1234567	**Message**: Login Success. **Action:**Redirect to user home.	**Message**: Login Success. **Action:**Redirect to user home.
TC4	Email:abc@gmail.com Password: 12	**Error**: User not found. Login failed.	**Error**: User not found. Login failed.
TC5	Email: Password: 12	**Error**: Email Required.	**Error**: Email Required.
TC6	Email:a@gmail.com Password: kartik	**Error**: Password Incorrect Login failed.	**Error**: Password Incorrect Login failed.

Table 3. Testing of a_login Functionality for user

4.4.2 Login Functionality Testing (Admin)

Test case no.	Case	Expected Outcome	Actual Outcome
TC1	Email: admin@how.com Password: 123@123	**Error**: Invalid Credentials. Login failed.	**Error**: Invalid Credentials. Login failed.
TC2	Email: admin@how.com Password:	**Error**: Password required.	**Error**: Password required.
TC3	Email: admin@how.com Password: 1234567	**Message**: Login Success. **Action:** Redirect to admin dashboard.	**Message**: Login Success. **Action:** Redirect to admin dashboard.

TC4	Email:abc@gmail.com Password: 12	**Error**: User not found. Login failed.	**Error**: User not found. Login failed.
TC5	Email: Password: 12	**Error**: Email Required.	**Error**: Email Required.
TC6	Email: admin@how.com Password: Kartik	**Error**: Password Incorrect Login failed.	**Error**: Password Incorrect Login failed.

Table 4. Testing of login functionality admin

4.4.3 Web App Functionality Testing (Logged in user)

Test case no.	**Case**	**Expected Outcome**	**Actual Outcome**
TC1	**Page:** User home **Action**: Button Click **Button**: Play guess game.	Redirect to Game Page and game countdown should start	Redirect to Game Page and game countdown should start
TC2	**Page:** User home **Action**: Button Click **Button**: Go to your profile.	Redirect to User profile Page	Redirect to User profile Page
TC3	**Page:** User Profile **Action**: Button Click **Button**: Update Feedback.	Feedback table is updated and reflected on page	Feedback table is updated and reflected on page
TC4	**Page:** User Profile **Action**: Button Click **Button**: Update name.	Name in user table is updated and reflected on page	Name not updated
TC5	**Page:** User Profile	Rank and score are fetched from backend and displayed	Rank and score are fetched from backend and displayed
TC6	**Page:** Home Page	Leaderboard table is populated from backend data from leaderboard table	Leaderboard table is populated from backend data from leaderboard table
TC7	**Page:** Home Page	Word of the day should be displayed with	Word of the day should be displayed with meaning

		meaning	
TC8	**Button:** Contact **Action:** Click	Redirect to contact page	Redirect to contact page
TC9	**Button:** Home **Action:** Click	Redirect to Home page	Redirect to Home page
TC10	**Button:** Go to your home page **Action:** Click	Redirect to user home page	Redirect to user home page
TC11	**Page:** Game **Action**: Matrix column click.	Character inside column should be filled up in answer field	Character inside column is filled up in answer field.
TC12	**Page:** Game **Action**: Button Click **Button**: Reset.	Level starts again. Timer restarts. Answer field Is cleared.	Level starts again. Timer restarts. Answer field Is cleared.
TC13	**Page:** Game **Action**: Button Click **Button**: Skip question.	Next question is displayed.	Nothing happens. **Error**: Render error
TC14	**Page:** Game **Action**: Button Click **Button**: Clear.	Answer field is cleared	Answer field is cleared
TC15	**Page:** Game	Matrix is populated with random alphabets	Matrix is populated with random alphabets
TC16	**Page:** Game **Action**: Button Click **Button**: Help.	Redirected to help page	Redirected to help page
TC17	**Page:** Game **Action**: Button Click **Button**: Skip question.	Next question is displayed.	Next question is displayed.

Table 5. Testing web app functionality.

4.5 Output at various stage

4.5.1 The Admin Dashboard Page:

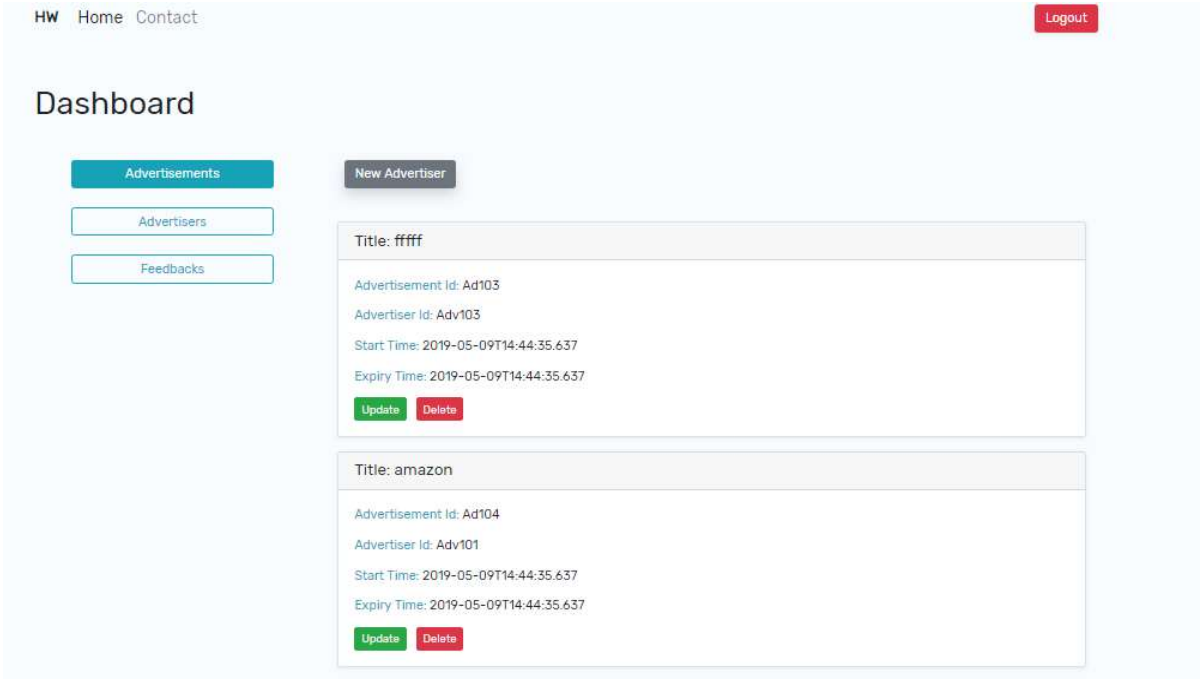


Fig *xvii.*

4.5.2 Add Advertiser Page

Fig *xviii.*

4.5.3 Contact Us Page

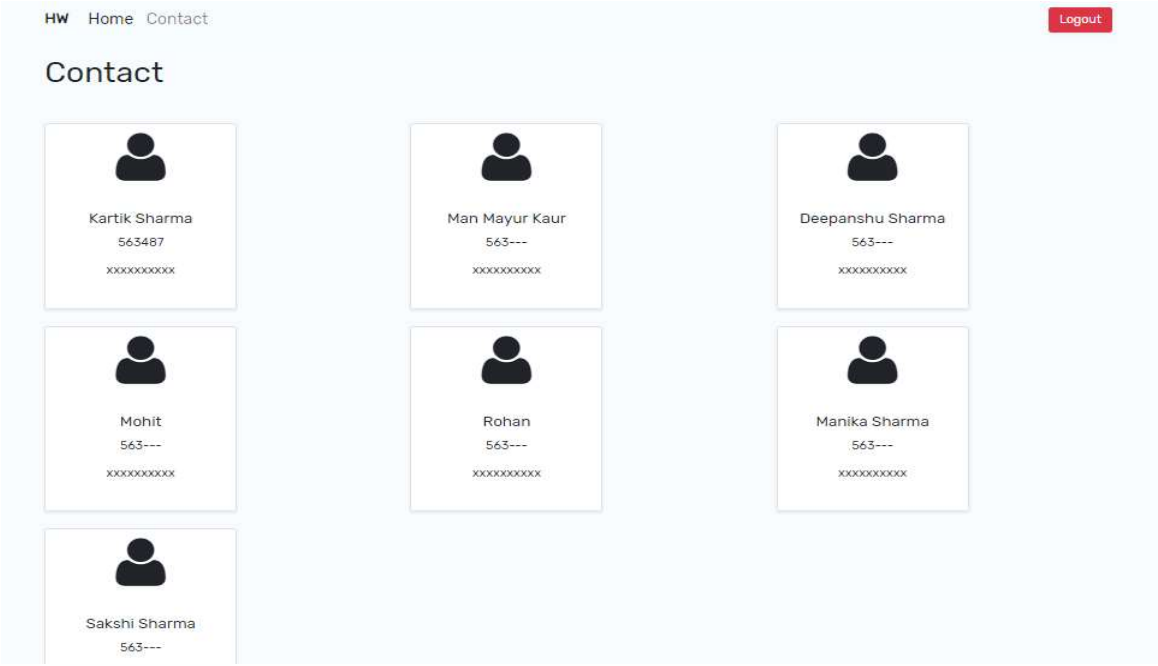


Fig *xix.*

4.5.4 Game Page

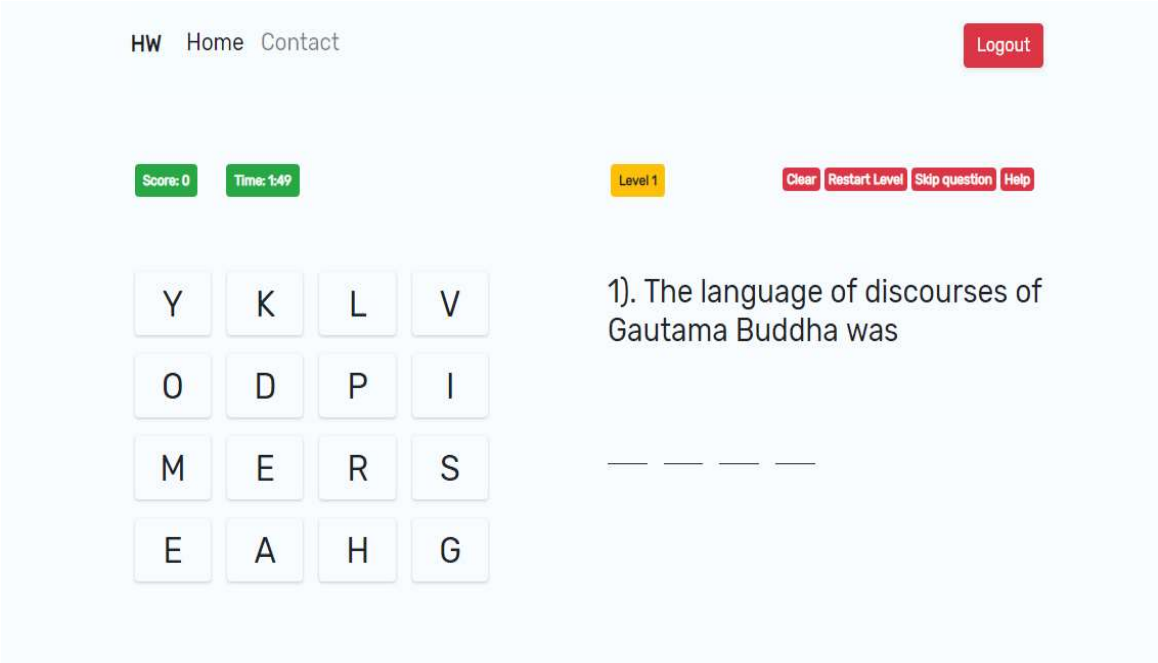


Fig *xx.*

4.5.5 Select Genre Page

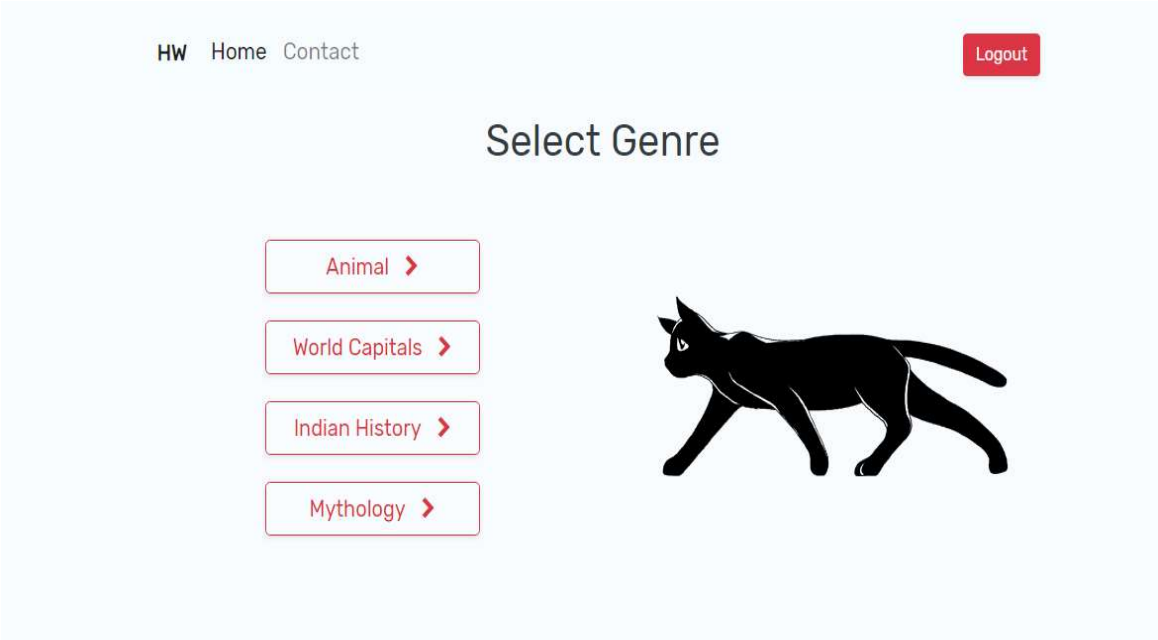


Fig *xxi.*

4.5.6 Help Page

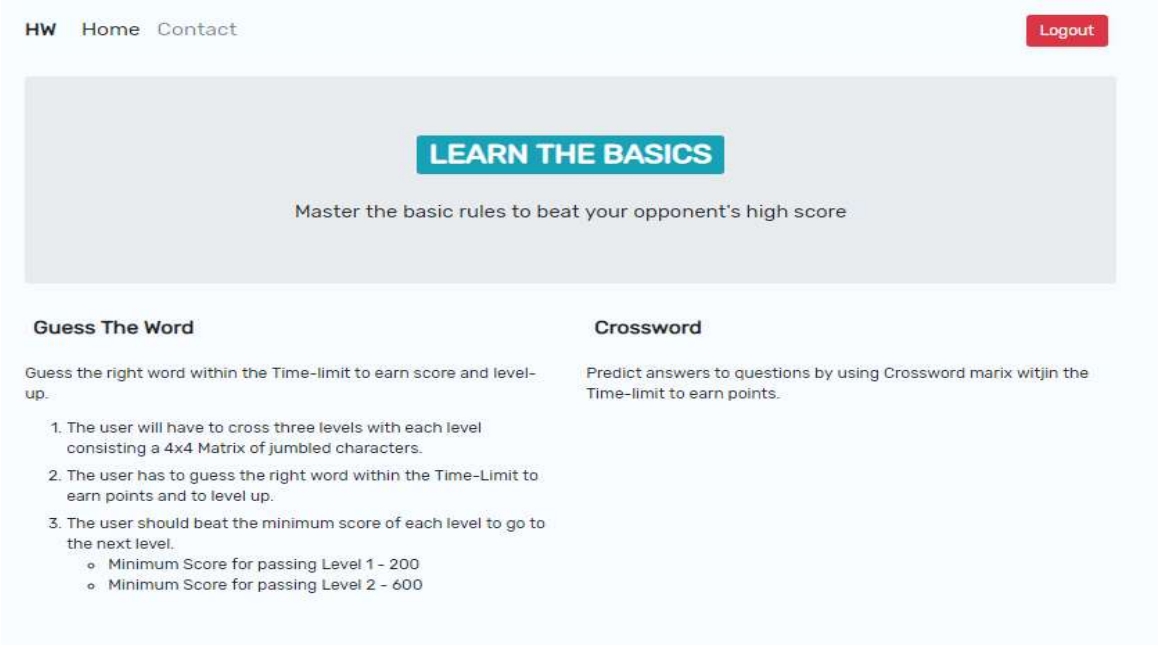


Fig *xxii.*

4.5.7 Login Page

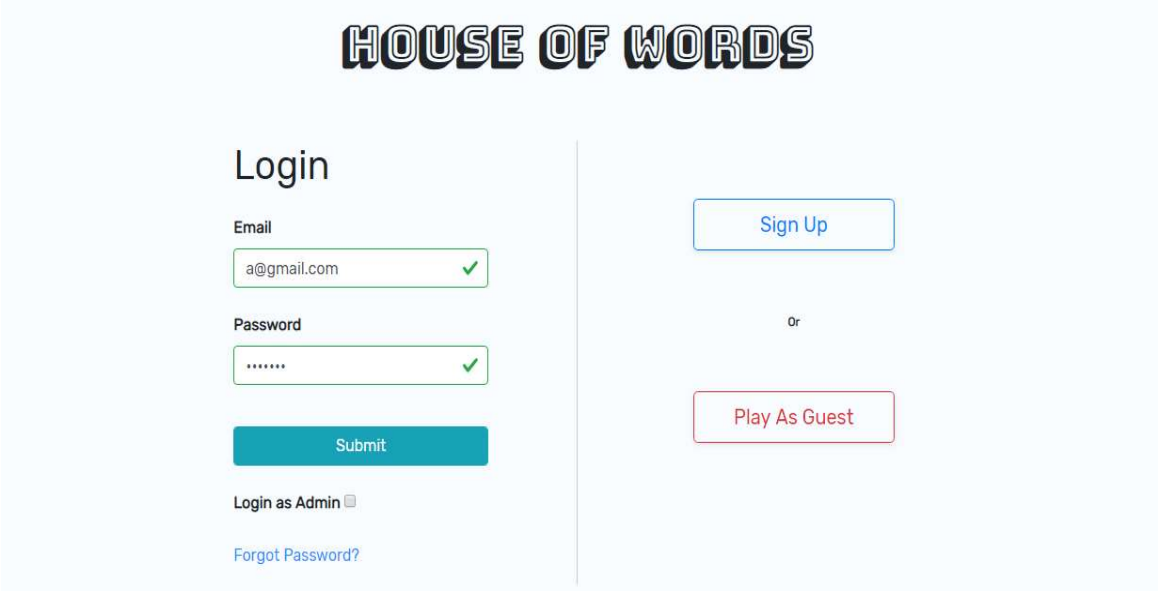


Fig *xxiii.*

4.5.8 Leaderboard Page

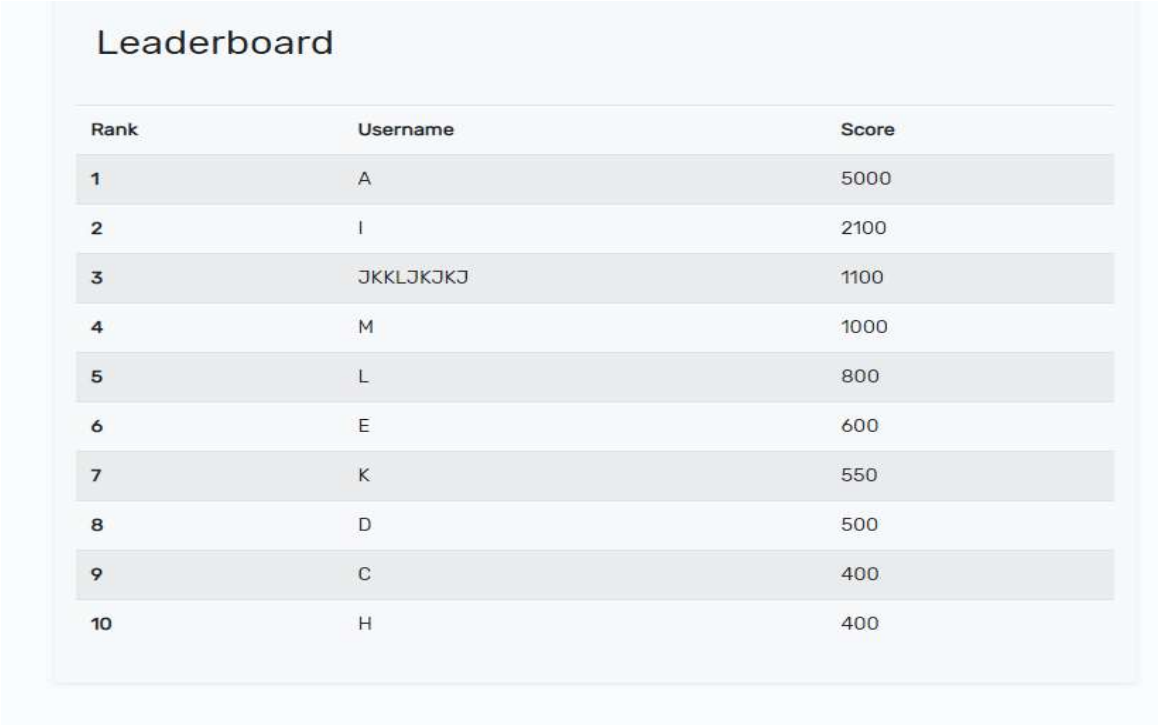


Fig *xxiv.*

1.4.9 Time Up Page

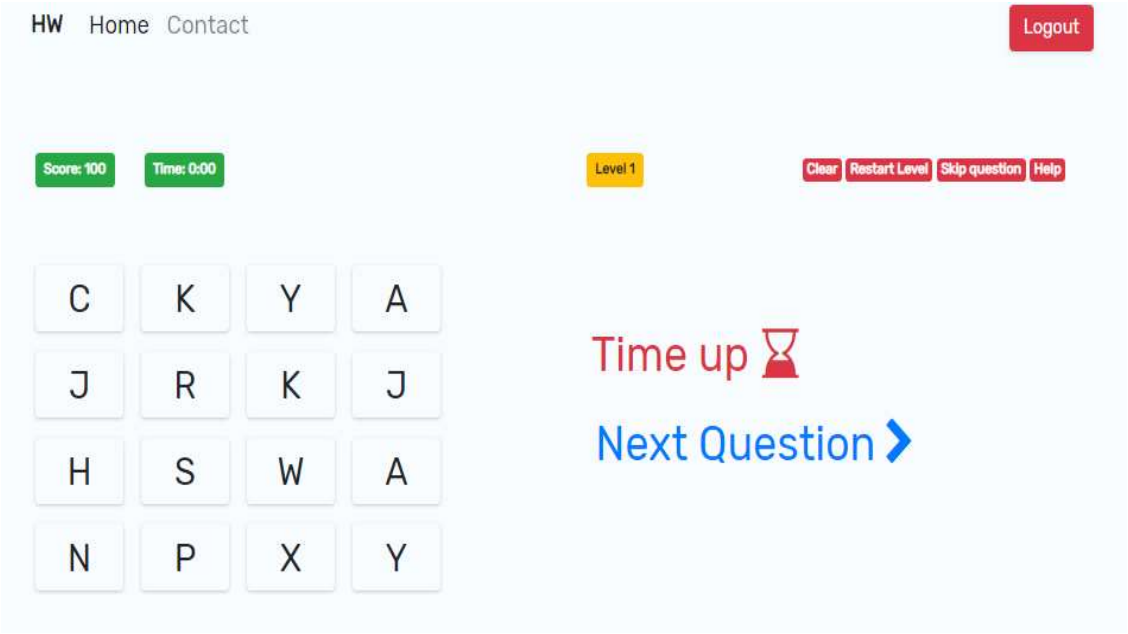


Fig *xxv.*

4.4.10 Update Feedback Page

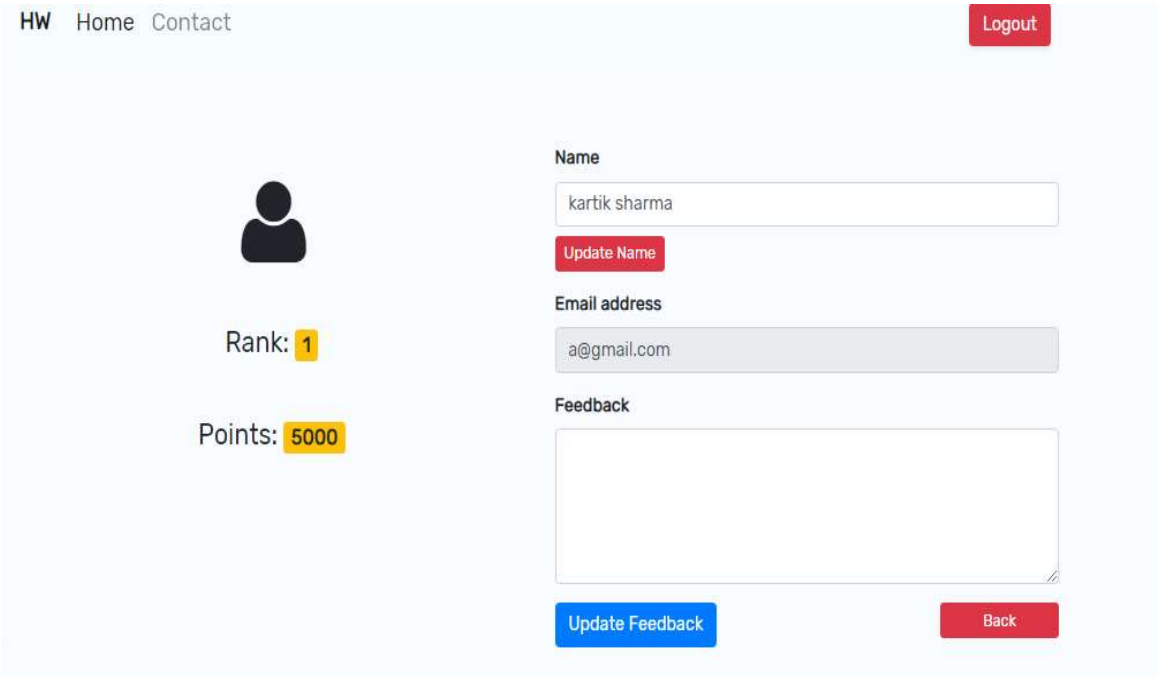


Fig *xxvi.*

4.4.11 User Home Page

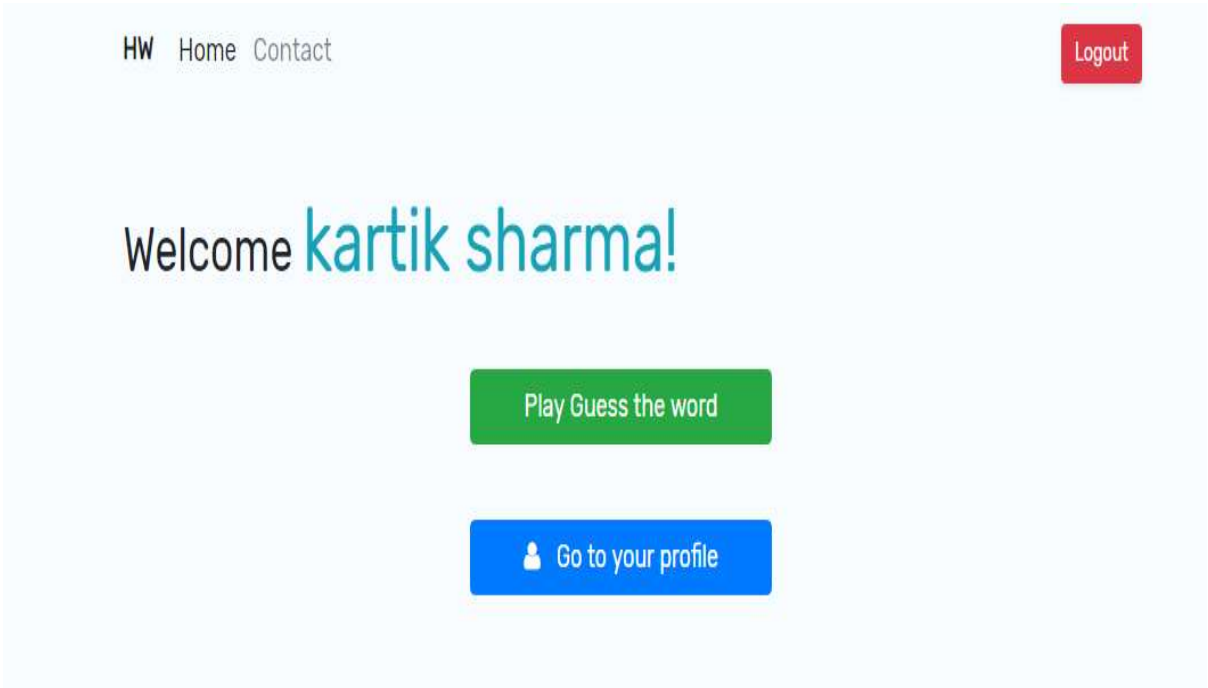


Fig *xxvii.*

4.4.12 Home Page

Fig *xxviii.*

4.4.13 Game Stats Modal

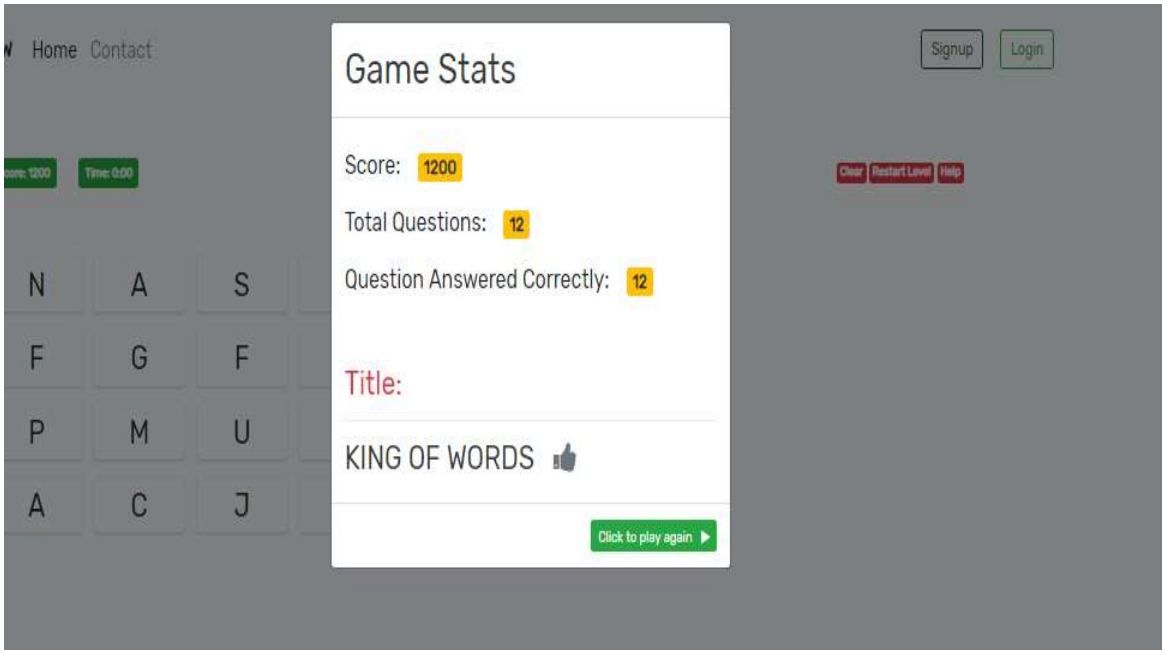


Fig *xxix.*

4.4.14 Reset Password

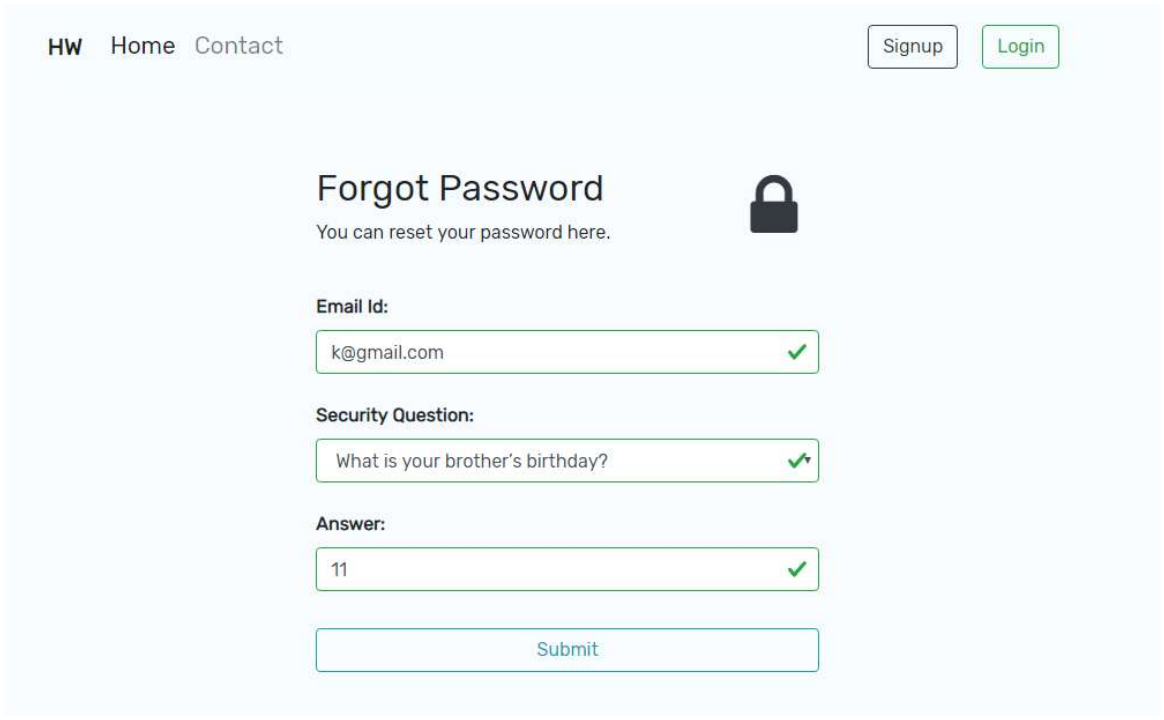


Fig *xxx.*

4.4.15 Feedbacks

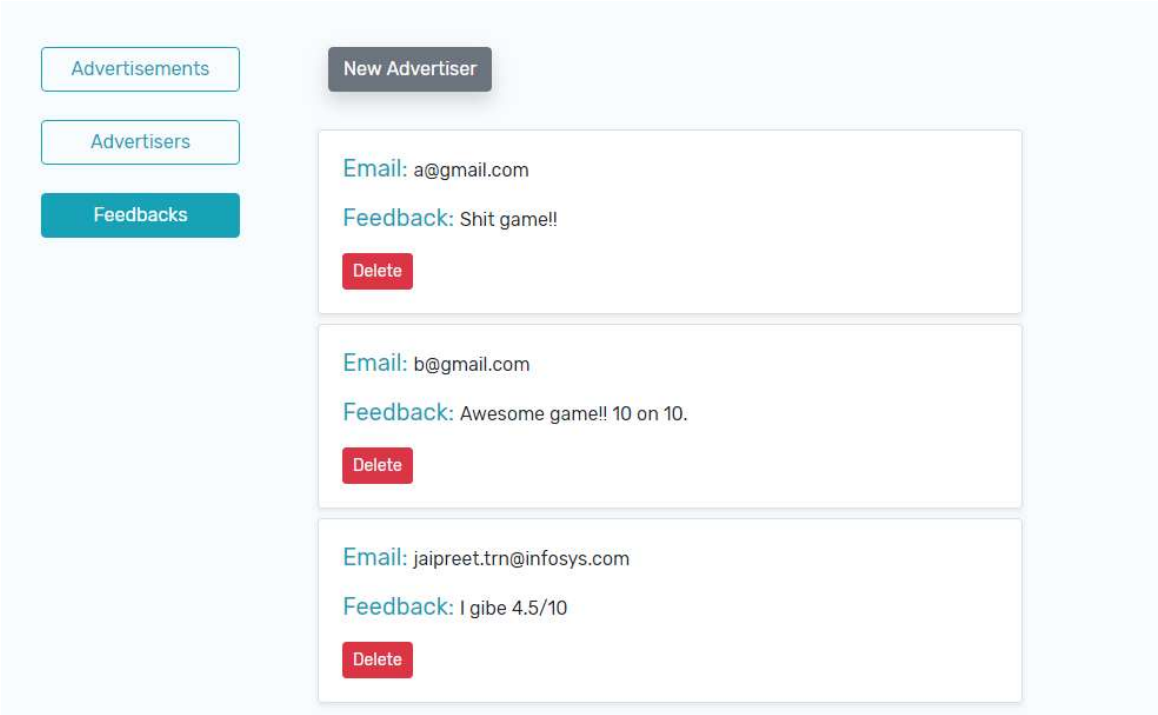


Fig *xxxi.*

4.4.16 Add Advertiser

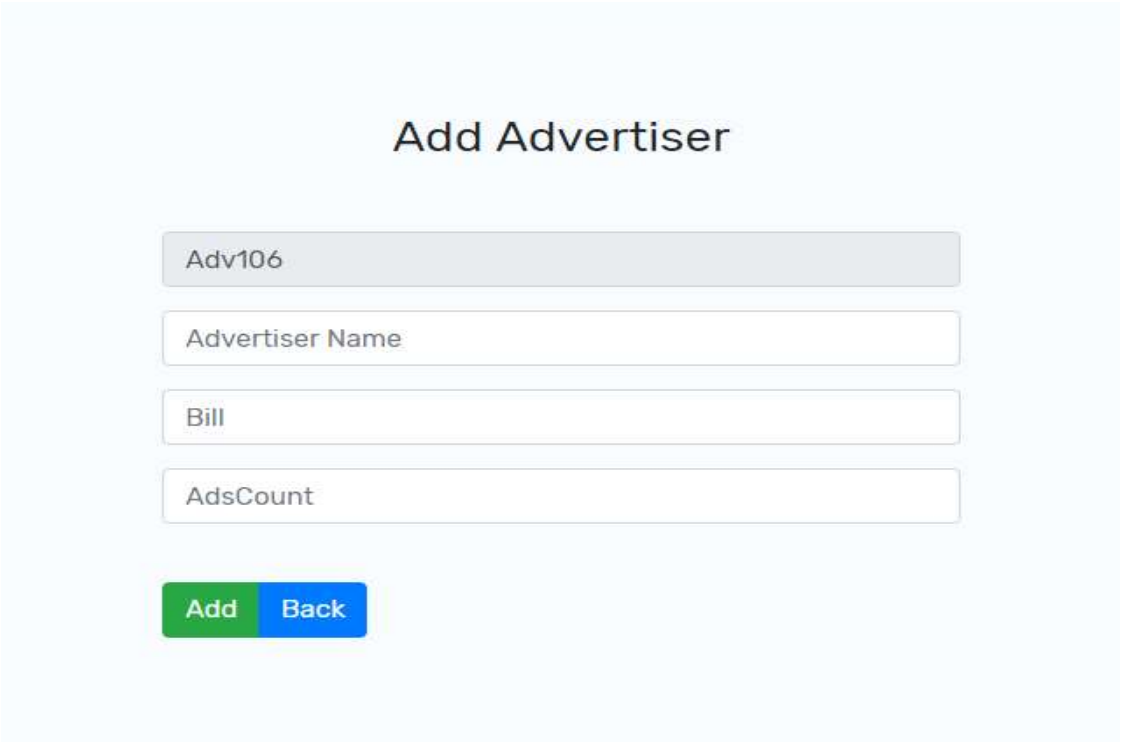


Fig *xxxii.*

CHAPTER 5 CONCLUSION

In the final phase, we conclude that the application satisfies all the expectations of the client and implements all the derived functionalities. It is debugged carefully to make sure that it results in no errors as well as exceptions. This application displays the mechanism and the various process that depends on the Model-View-Controller architecture and Vue.Js that provides a better view to the project in creating a more of a what styled display. The dependency between the backend and the front-end provided by the API add to the ease and a proper functionality of the project.

5.1 FUTURE SCOPE

The game designed will have a scope for improvement. As it is a game, there is no stopping to it. More and more functionalities can be added to it and it is always open to more and more complex matrixes that can be added to improve it. The design can be integrated with a variety of games thus providing more and more user interactions**.**

The application can be further enhanced in a variety of ways:

1.The designs can be further enhanced to make it look more attractive and better-looking.

2.Account Locking functionalities can be implemented to block accounts using machine-learning to block all those accounts involving a single user with multiple accounts, use of irrelevant remarks and more.

3.Email notifications can be sent to users machines to brief them about the various gaming challenges and competitions.

4. Any unwanted breaches and hacks can be reported to the admin and can be dealt with immediately.

REFERENCES

[1] http://www.w3schools.com

[2] DesignandImplementationofanMVC-BasedArchitectureforE-CommerceApplications by E. Althammer and W. Pree Published date 5th september2013.

[3] Assessing the Effectiveness of the Model View Controller Architecture for Creating Web Applications Authors: Nick Heidke, Joline Morrison, and Mike Morrison Department of Computer Science University of Wisconsin-EauClaire

[4] The Progressive JavaScript Framework, https://vuejs.com

[5] Bootstrap, https://getbootstrap.com/

[6] Evaluation of the Model-View-Controller design pattern when applied to a heterogeneous application to distribute newspaper textual content to mobile devices", SakibSupple

[7] MVC Design Pattern and ASP .NET MVC Framework Research [J]J LU, L CHANG, Y CHEN - Computer Knowledge and Technology, 2010 - en.cnki.com.cn

[8] GuangChun, L., WangYanhua, Xianliang, L, and Hanhong. A Novel Web Application Frame Developed by MVC. Software Engineering Notes. 2003, Vol. 28,2.

[9] Leff, A. and Rayfield, J.T. Web-Application Development Using the Model/View/Controller Design Pattern. IEEE XPlore.2001.

[10] Sauter, P., Vogler, G., Specht, G., and Flor, T. A Model-View-Controller

extension for pervasive multi-client user interfaces. Pers Ubiquit Comput. October,2004.

[11] Open MVC: A Non-proprietary Component-based Framework for Web Applications Barrett, R. and Delany, S.J. New York: ACM, WWW2004.

[12] ADatabaseandWebApplicationBasedonMVCArchitecture.Selfa,D.M.,Carrillo,M., andRocioBoone,M.Puebla,Mexico:IEEE,IEEEInt.Conf.onElectronics,Communications, and Computers (CONIELECOMP2006).

[13] Domain Driven Web Development With WebJinn. Kojarski, S. and Lorenz, D.H. Anaheim, CA : ACM, OOPSLA2003.

[14] Apache Struts 2 Documentation http://struts.apache.org/2.0.6/docs/home.html.2/12/2007

[15] MVC-based Architecture for e-commerce.Journal.doc22/22

[16] R. Eckstein, M. Loy, D. Wood, Java Swing (O'Reilly,1998)

[17] M. Fontoura, W. Pree, B. Rumpe, The UML Profile of Framework Architectures (Addison-Wesley, 2000)

[18] C. Hewitt, Developing Business Object-based Applications in JBuilder,1998

[19] The Apache Software Foundation. Apache Struts Web Application Framework. http://jakarta.apache.org/struts.

[20] Maverick Project. Source Forge. http://mav.sourceforge.net.

[21] M. Campione, K. Walrath, The Java Tutorial Second Edition. Object-Oriented Programming for the Internet (Addison Wesley,1999)