"BUILDOUTS"

"USING SPRING REQUESTMAPPING"

A

Project Report

submitted

in partial fulfillment

for the award of the Degree of

Bachelor of Technology

in Department of Computer Science and Engineering

Project Mentor:
Name: Dr. Niketa Sharma
Designation : Associate Professor

Submitted By :
Jeevandeep Singh, 17ESKCS075
Kavish Gupta, 17ESKCS078
Prince Jain, 17ESKCS121

Department of Computer Science and Engineering
Swami Keshvanand Institute of Technology, M & G, Jaipur
Rajasthan Technical University, Kota
Session 2020-2021

Swami Keshvanand Institute of Technology, Management & Gramothan, Jaipur

Department of Computer Science and Engineering

CERTIFICATE

This is to certify that Mr. Prince Jain, Mr. Kavish Gupta and Mr. Jeevan Deep Singh a student of B.Tech(Computer Science & Engineering) 7th semester has submitted his/her Project Report entitled ”BUILDOUTS (Using Spring RequestMapping)” under my guidance.

Mentor

Name: Dr. Niketa Sharma

Designation: Associate Professor

Signature.....

Coordinator

Name: Dr. Mukesh Gupta &

Mrs. Anjana Sangwan

Designation.....

Signature.....

DECLARATION

We hereby declare that the report of the project entitled ”BUILD-OUTS” is a record of an original work done by us at Swami Keshvanand Institute of Technology, Management and Gramothan, Jaipur under the mentorship of ”Dr. Niketa Sharma” (Dept. of Computer Science and Technology) and coordination of ”Dr. Mukesh Gupta & Mrs. Anjana Sangwan” (Dept.of Computer Science and Technology). This project report has been submitted as the proof of original work for the partial fulfillment of the requirement for the award of the degree of Bachelor of Technology (B.Tech) in the Department of Computer Science and Technology.It has not been submitted anywhere else, under any other program to the best of our knowledge and belief.

Team Members

Signature

Jeevandeep Singh, 17ESKCS075

Kavish Gupta, 17ESKCS078

Prince Jain, 17ESKCS121

Acknowledgement

A project of such a vast coverage cannot be realized without help from numerous sources and people in the organization.We take this opportunity to express our gratitude to all those who have been helping us in making this project successful.

We are highly indebted to our faculty mentor Dr. Niketa Sharma. She has been a guide, motivator source of inspiration for us to carry out the necessary proceedings for the project to be completed successfully. We also thank our project coordinator Dr. Mukesh Gupta & Mrs. Anjana Sangwan for his/her co-operation, encouragement, valuable suggestions and critical remarks that galvanized our efforts in the right direction.

We would also like to convey our sincere thanks to Prof. Dr. Mukesh Gupta, HOD, Department of Computer Science and Engineering, for facilitating, motivating and supporting us during each phase of development of the project. Also, we pay our sincere gratitude to all the Faculty Members of Swami Keshvanand Institute of Technology, Management and Gramothan, Jaipur and all our Colleagues for their co-operation and support.

Last but not least we would like to thank all those who have directly or indirectly helped and cooperated in accomplishing this project.

Team Members:

Jeevandeep Singh, 17ESKCS075

Kavish Gupta, 17ESKCS078

Prince Jain, 17ESKCS121

Contents

List of Figures

Chapter 1

Introduction

1.1 Problem Statement and Objective

To build a Back-End service for Java Application. In this application we have to receive the user response in form of JSON data (questions and user's answers) from API call. The motive of this project is validation of a online Quiz, QA from and Generating the report of user response.

1.2 Definition

Standardize the data exchange format so that both the front-end and back-end can communicate in a language-independent manner.
Create a scalable Java Application to support multiple accessibility

1.3 Proposed Logic

Implement a Quiz API which helped to render the quiz on the UI and assess the answers. Creation of a Spring Boot server from scratch. Setup of port number as well as creation of SpringBoot Application file along with Properties file. Make database calls with Mongo-DB for CRUD operations using Spring

JPA. And perform Request Mapping(get, put, post) for storing and retrieving data from database.

1.4 Scope of the Project

Buildout is a Question and Answer form but it is more scalable because we use MVCS Architecture which helps us in providing Abstraction. We also use Java Interfaces so any changes in future will not effect whole Structure. We have used JSON which is language independent so in future we will not take care of from which client our application is used. We have also used Nosql database which is MongoDB so we are not focusing on specific structure of database and in future we can easily change it according to the client requirement This application is build on java which guarantees about security and efficiency. Also Created Unit Test which cover all edge cases so any update in future which can lead to error can easily detected.

Chapter 2

Software Requirement Specification

2.1 Overall Description

Buildouts is basically a Spring Boot Application. It is a Back-End service for simple Question-Answer (consist of MCQs) form. In this application we receive the user response in form of JSON data (questions and user's answers) from API call. Then we check the user's answers with correct answers present in Database and then return the data in form of response object in which there are correct answer, explanation for every question and final score. The importance of this project is validation of a online Quiz, QA from and Generating the report of user response.

2.1.1 Product Perspective

2.1.1.1 System Interfaces

The application runs in the latest version of Chrome or Firefox browser on Windows, Linux and Mac.

2.1.1.2 User Interfaces

A simple Question Answer Form That can accept user inputs from

1. Check Box
2. Radio Button
3. Textbox
4. Submit Button

2.1.1.3 Hardware Interfaces

1. 1TB hard disk space in Server Machine.
2. RAM: 4GB
3. Processor: Core i5
4. 20GB of hard disk space in terminal machines

2.1.1.4 Software Interfaces

1. Windows above than (Version 7) / Linux (ubuntu, mac)
2. Latest version of Chrome (8.1 or above) or another Browser.]
3. Android version 4.0 or higher to perform.

2.1.1.5 Communications Interfaces

1. The communication architecture must follow the client-server model.

2. Communication between the client and server should utilize a REST-compliant web service and must be served over HTTP Secure (HTTPS).

3.The client-server communication must be stateless. A uniform interface must separate the client roles from the server roles.

2.1.1.6 User Characteristics

Admin → He can use all CRUD operations like create, read, update and delete on database using request-mapping. Admin can update set of new questions into DB, He can also change the marking pattern as well. He can also provide the report of exam to user.
User → He can use only read operation to access the form and submit his response using post-mapping.

2.1.1.7 Constraints

1. Users have to download the application on their phone in order to use it.
2. There should be at least 3 MB of free space on the device.
3. Java should be the programming language used in implementation with SpringBoot.

2.1.1.8 Assumption and Dependencies

1. It is assumed that the user should familiar with Computer Device having internet connection in the device.
2. It is assumed that all access has given to Admin only.
3. Initially Question list should be in a Database.

4. Server should be maintained

5. Front-end has been already Developed.

Chapter 3

SYSTEM DESIGN SPECIFICATION

3.1 System Architecture

MVCS Architecture: At a broader level, clients initiate requests to the server. These requests adhere to REST API principles and are sent to the server using the HTTP protocol.When the request reaches the server, a response is triggered in 3 stages:

Controller layer: The request is intercepted by a controller that directs it to the service layer.

Service layer: The specification of the request is understood by the business logic function and sent to the next stage for data retrieval.

Repository layer: Based on the requirements, appropriate data is retrieved from the database and returned to the service layer. In a similar way, the expected response is returned to the client.

3.2 Technologies Used Module

3.2.1 JAVA

Java is a programming language that produces software for multiple platforms. When a programmer writes a Java application, the

compiled code (known as byte code) runs on most operating systems (OS), including Windows, Linux and Mac OS. Java derives much of its syntax from the C and C++ programming languages.

Java produces applets (browser-run programs), which facilitate graphical user interface (GUI) and object interaction by Internet users. Prior to Java applets, Web pages were typically static and non-interactive. Java applets have diminished in popularity with the release of competing products, such as Adobe Flash and Microsoft Silver light.

Java applets run in a Web browser with Java Virtual Machine (JVM), which translates Java byte code into native processor instructions and allows indirect OS or platform program execution. JVM provides the majority of components needed to run byte code, which is usually smaller than executable programs written through other programming languages. Byte code cannot run if a system lacks required JVM.

Java program development requires a Java software development kit (SDK) that typically includes a compiler, interpreter, documentation generator and other tools used to produce a complete application.

Development time may be accelerated through the use of integrated development environments (IDE) - such as JBuilder, Netbeans, Eclipse or JCreator. IDEs facilitate the development of GUIs, which include buttons, text boxes, panels, frames, scrollbars and other objects via drag-and-drop and point-and-click actions.

Java programs are found in desktops, servers, mobile devices, smart cards and Blu-ray Discs (BD).

3.2.2 SpringBoot FrameWork

Spring Boot is a brand new framework from the team at Pivotal, designed to simplify the bootstrapping and development of a new Spring application. The framework takes an opinionated approach to configuration, freeing developers from the need to define boilerplate configuration.

Prior to the advent of Enterprise Java Beans (EJB), Java developers needed to use JavaBeans to create Web applications. Although JavaBeans helped in the development of user interface (UI) components, they were not able to provide services, such as transaction management and security, which were required for developing robust and secure enterprise applications. The advent of EJB was seen as a solution to this problem EJB extends the Java components, such as Web and enterprise components, and provides services that help in enterprise application development. However, developing an enterprise application with EJB was not easy, as the developer needed to perform various tasks, such as creating Home and Remote interfaces and implementing lifecycle callback methods which lead to the complexity of providing code for EJBs Due to this complication, developers started looking for an easier way to develop enterprise applications.

The Spring framework has emerged as a solution to all these complications This framework uses various new techniques such as Aspect-Oriented Programming (AOP), Plain Old Java Object (POJO), and de-

pendency injection (DI), to develop enterprise applications, thereby removing the complexities involved while developing enterprise applications using EJB, Spring is an open source lightweight framework that allows Java EE 7 developers to build simple, reliable, and scalable enterprise applications. This framework mainly focuses on providing various ways to help you manage your business objects. It made the development of Web applications much easier as compared to classic Java frameworks and Application Programming Interfaces (APIs), such as Java database connectivity(JDBC), JavaServer Pages(JSP), and Java Servlet. But still lot of configration part.

We can choose Spring Boot because of the features and benefits it offers as given here

1.It provides a flexible way to configure Java Beans, XML configurations, and Database Transactions.

2.It provides a powerful batch processing and manages REST endpoints.

3.In Spring Boot, everything is auto configured; no manual configurations are needed.

4.It offers annotation-based spring application

5.Eases dependency management

6.It includes Embedded Servlet Container

3.3 Rest API

REpresentational State Transfer (REST) is an architectural style that defines a set of constraints to be used for creating web services. REST API is a way of accessing the web services in a simple and flexible way without having any processing.

REST technology is generally preferred to the more robust Simple Object Access Protocol (SOAP) technology because REST uses the less bandwidth, simple and flexible making it more suitable for internet usage. It's used to fetch or give some information from a web services. All communication done via REST API used only HTTP request.

3.3.1 Working

A request is send from client to server in the form of web URL as HTTP GET or POST or PUT or DELETE. After that a response is come back from server in the form of resource which can be anything like HTML, XML, Image or JSON. But now JSON is the most popular format being used in Web Services.

In HTTP there are five methods which are commonly used in a REST based Architecture i.e., POST, GET, PUT, PATCH, and DELETE. These correspond to create, read, update, and delete (or CRUD) operations respectively. There are other methods which are less frequently used like OPTIONS and HEAD.

3.3.2 GET

The HTTP GET method is used to read (or retrieve) a representation of a resource. In the safe path, GET returns a representation in XML or JSON and an HTTP response code of 200 (OK). In an error case, it most often returns a 404 (NOT FOUND) or 400 (BAD REQUEST).

3.3.3 POST

The POST verb is most-often utilized to create new resources. In particular, it's used to create subordinate resources. That is, subordinate to some other (e.g. parent) resource. On successful creation, return HTTP status 201, returning a Location header with a link to the newly-created resource with the 201 HTTP status. NOTE: POST is neither safe nor idempotent.

3.3.4 PUT

It is used for updating the capabilities. However, PUT can also be used to create a resource in the case where the resource ID is chosen by the client instead of by the server. In other words, if the PUT is to a URI that contains the value of a non-existent resource ID. On successful update, return 200 (or 204 if not returning any content in the body) from a PUT. If using PUT for create, return HTTP status 201 on successful creation. PUT is not safe operation but it's idempotent.

3.4 High Level Design Diagrams

3.4.1 Use Case Diagram

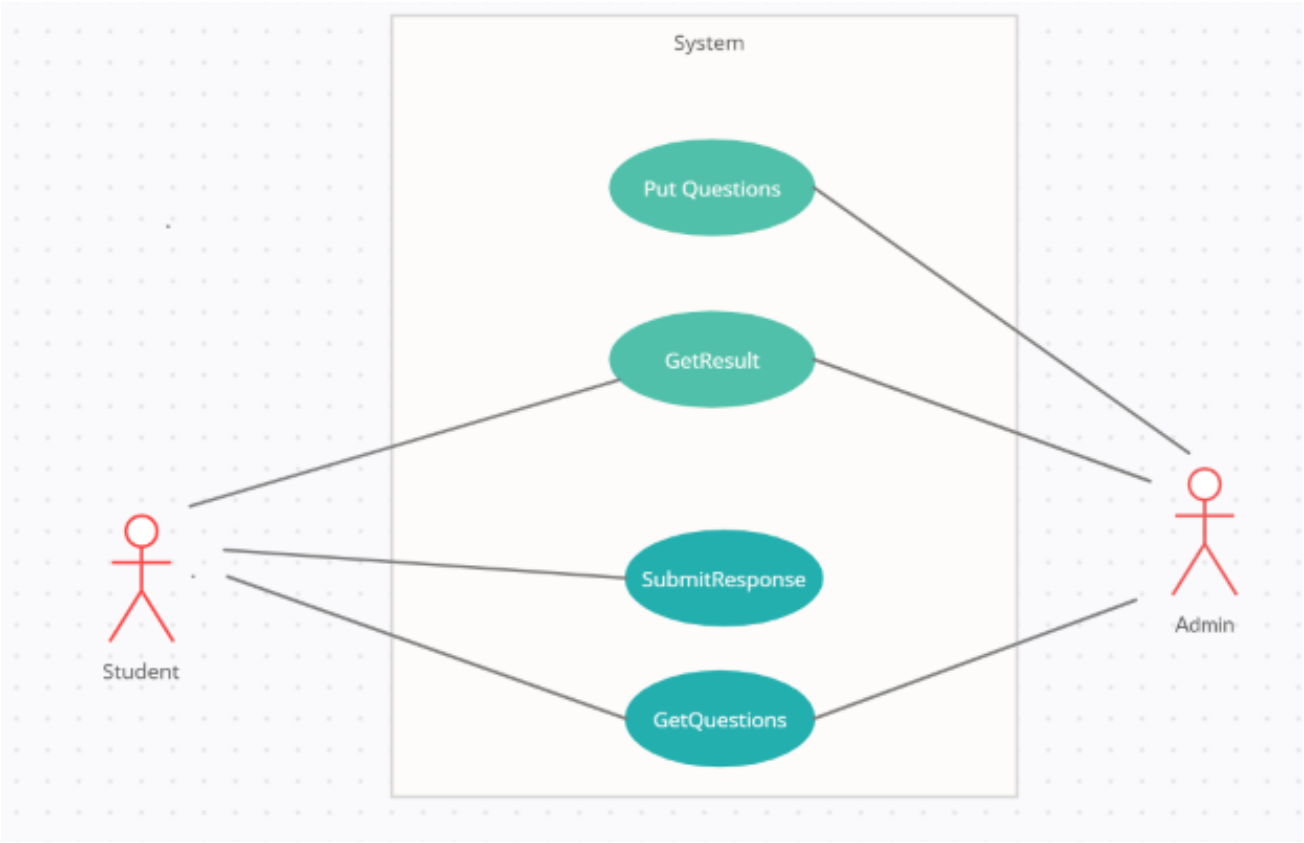

Figure 3.1: Use Case Diagram

3.4.2 Activity Diagram

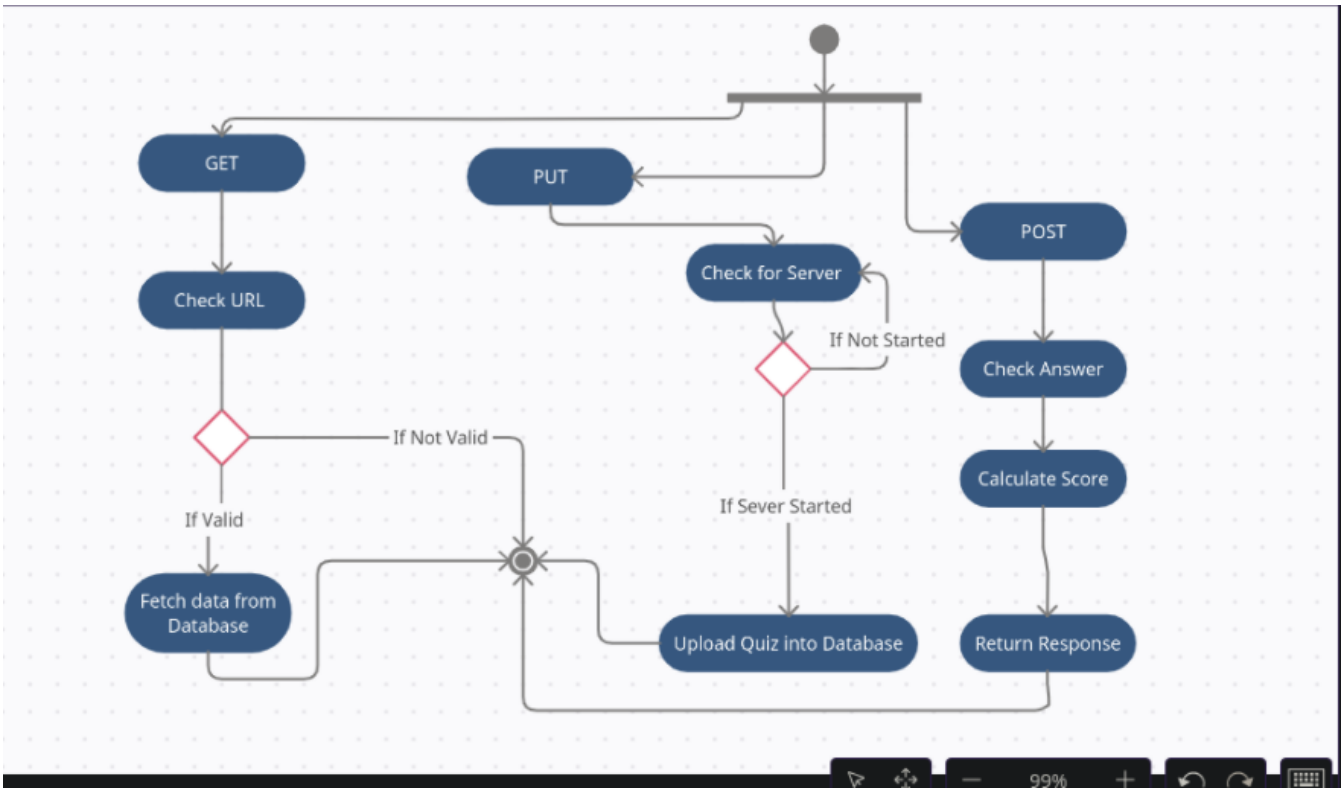

Figure 3.2: Activity Diagram

3.4.3 Data-Flow Diagram

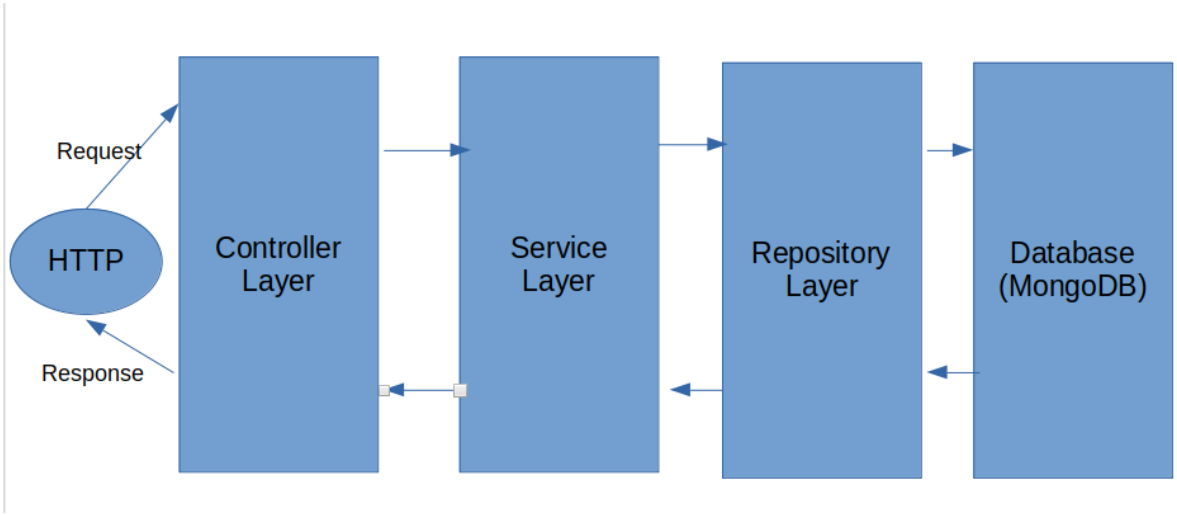

Figure 3.3: Data-Flow Diagram

3.4.4 Class Diagram

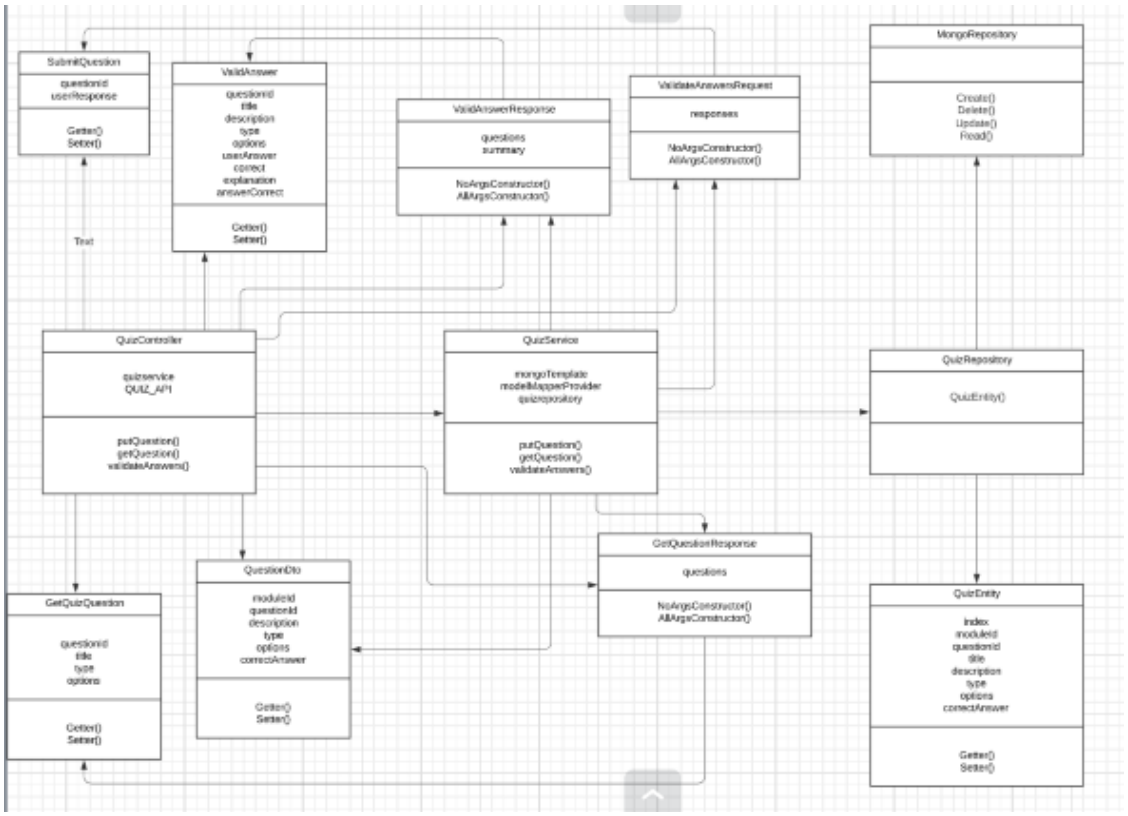

Figure 3.4: Class Diagram

Chapter 4

METHODOLOGY AND TEAM

4.1 Introduction to Waterfall Framework

The Waterfall Model was first Process Model to be introduced. It is also referred to as a linear-sequential life cycle model. It is very simple to understand and use. In a waterfall model, each phase must be completed before the next phase can begin and there is no overlapping in the phases. The waterfall Model illustrates the software development process in a linear sequential flow; hence it is also referred to as a linear-sequential life cycle model. This means that any phase in the development process begins only if the previous phase is complete. In waterfall model phases do not overlap. In ”The Waterfall” approach, the whole process of software development is divided into separate phases. In Waterfall model, typically, the outcome of one phase acts as an input for the next phase sequentially. Following is a diagrammatic representation of different phases of waterfall model.

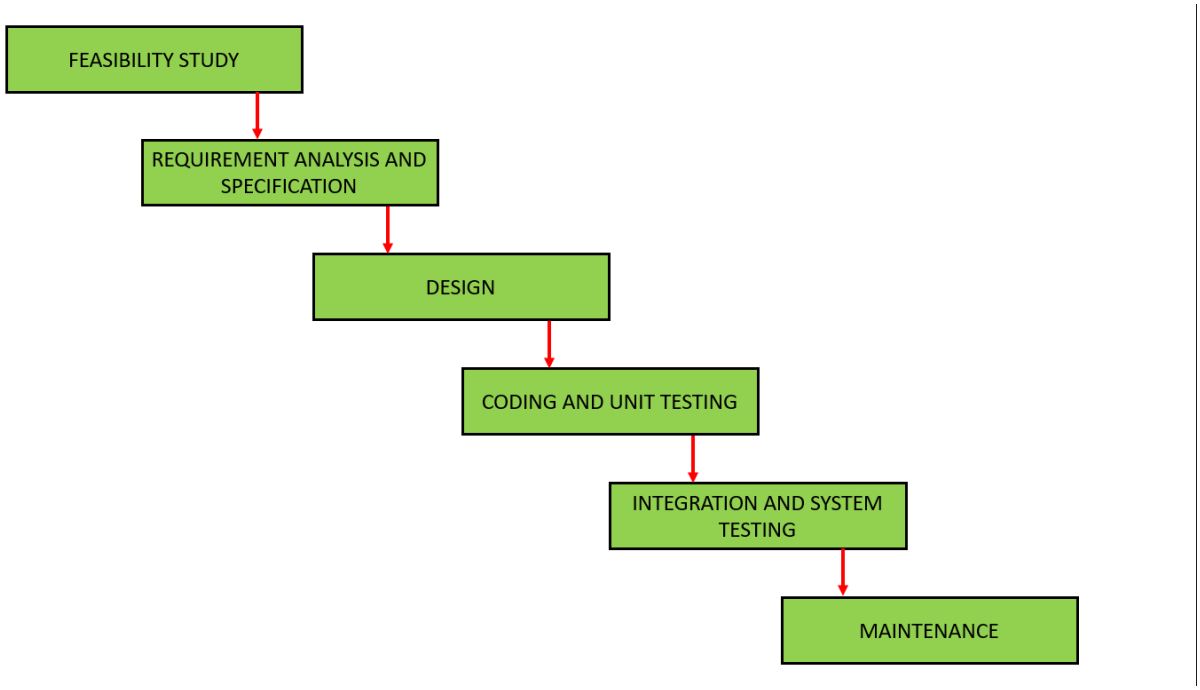

Figure 4.1: WaterFall model

The sequential phases in Waterfall model are-

1. **Requirement Gathering and analysis:** All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification doc.

2. **System Design:** The requirement specifications from first phase are studied in this phase and system design is prepared. System Design helps in specifying hardware and system requirements and also helps in defining overall system architecture.

3. **Implementation:** With inputs from system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality which is referred to as Unit Testing.

4. **Integration and Testing:** All the units developed in the imple-

mentation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.

5. **Deployment of system:** All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.

6. **Maintenance:** All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.

All these phases are cascaded to each other in which progress is seen as flowing steadily downwards (like a waterfall) through the phases. The next phase is started only after the defined set of goals are achieved for previous phase and it is signed off, so the name ”Waterfall Model”. In this model phases do not overlap.

Waterfall Model Pros Cons

Advantage The advantage of waterfall development is that it allows for departmentalization and control. A schedule can be set with deadlines for each stage of development and a product can proceed through the development process model phases one by one. Development moves from concept, through design, implementation, testing, installation, troubleshooting, and ends up at operation and maintenance. Each phase of development proceeds in strict order.

Disadvantage The disadvantage of waterfall development is that it does not allow for much reflection or revision. Once an application is in the testing stage, it is very difficult to go back and change something that was not well-documented or thought upon in the concept stage.

4.2 Team Members, Roles & Responsibilities

Jeevan deep Singh- Database
Kavish Gupta - Product Design
Prince Jain - Scripting and testing

Chapter 5

System Testing

The designed system has been testing through following test parameters.

5.1 Functionality Testing

In testing the functionality of the web sites the following features were tested:

1. Links

   (a) Internal Links: All internal links of the website were checked by clicking each link individually and providing the appropriate input to reach the other links within.

   (b) External Links: Till now no external links are provided on our website but for future enhancement we will provide the links to the candidate's actual profile available online and link up with the elections updates online etc.

   (c) Broken Links : Broken links are those links which so not divert the page to specific page or any page at all. By testing

the links on our website, there was no link found on clicking which we did not find any page.

2. Forms

   (a) Error message for wrong input : Error messages have been displayed as and when we enter the wrong details (eg. Dates), and when we do not enter any details in the mandatory fields. For example: when we enter wrong password we get error message for acknowledging us that we have entered it wrong and when we do not enter the username and/or password we get the messages displaying the respective errors.

   (b) Optional and Mandatory fields : All the mandatory fields have been marked with a red asterisk (*) and apart from that there is a display of error messages when we do not enter the mandatory fields. For example: As the first name is a compulsory field in all our forms so when we do not enter that in our form and submit the form we get an error message asking for us to enter details in that particular field.

3. Database Testing is done on the database connectivity.

5.2 Performance Testing

Integration Testing $\rightarrow$ Integration testing is the phase in software testing in which individual software modules are combined and tested as a group.

Unit Testing $\rightarrow$ Unit tests are typically automated tests written and run by software developers to ensure that a section of an application (known as the ”unit”) meets its design and behaves as intended.

Chapter 6

TEST EXECUTION SUMMARY

1. Controller Class Test Summary :

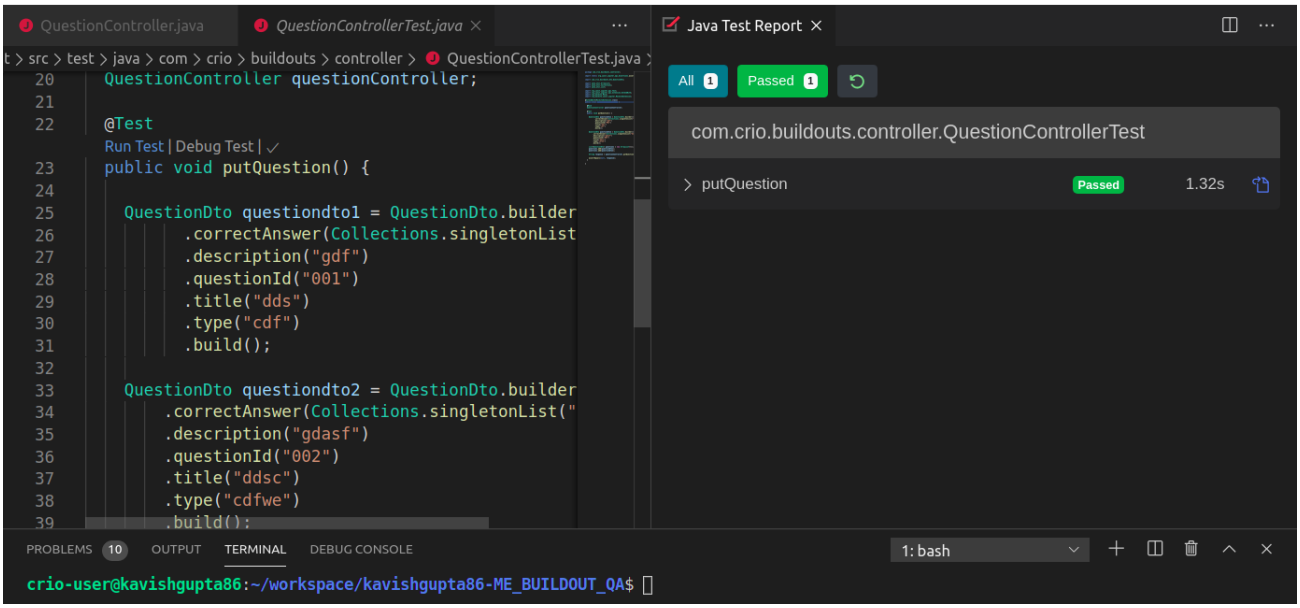

Figure 6.1: QuestionControllerTest.java

2. Service Class Test Summary :

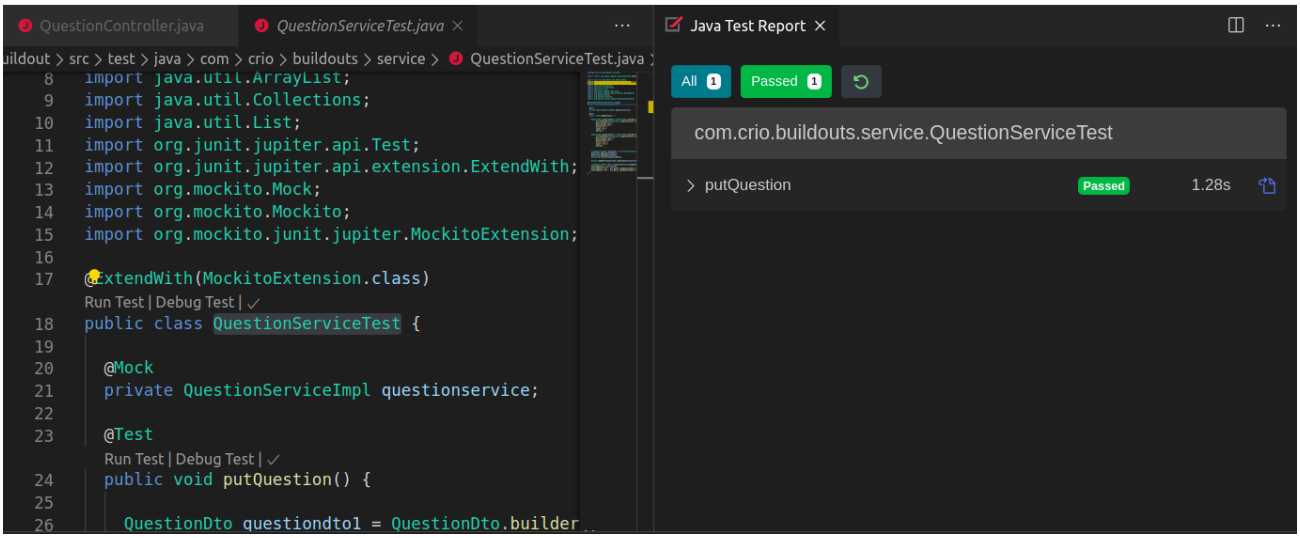

Figure 6.2: QuestionServiceTest.java

3. Dto Class Test Summary :

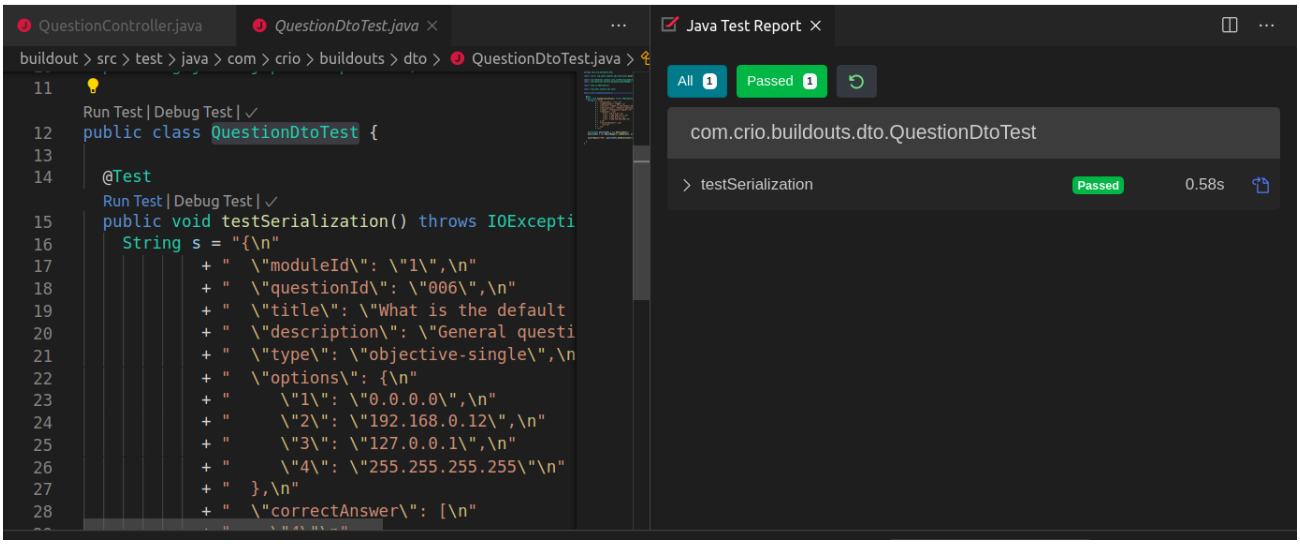

Figure 6.3: QuestionDtoTest.java

Chapter 7

PROJECT SCREENSHOTS

GET Method →
It is used to fetch the data from the Database

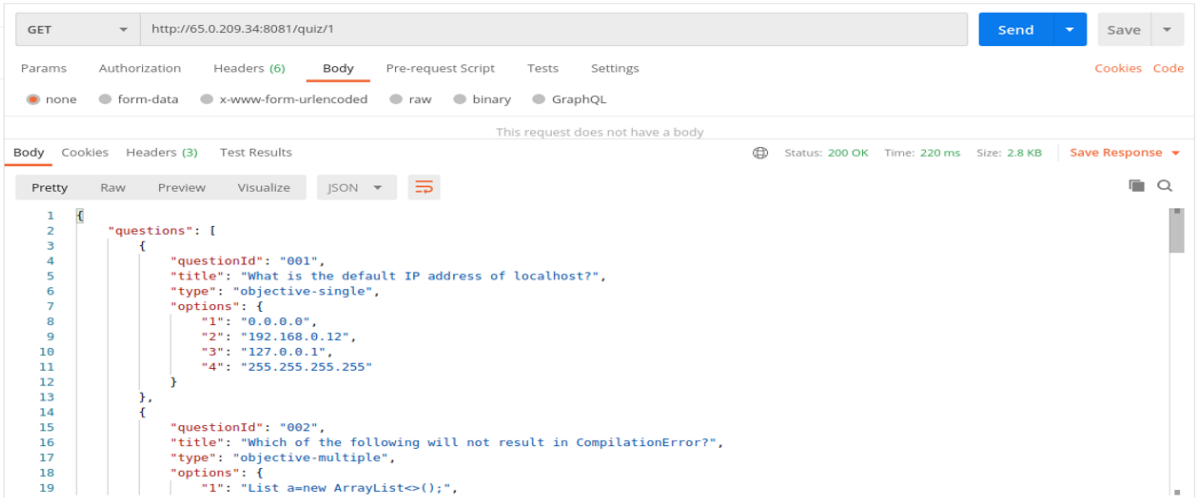

Figure 7.1: GET Method

POST Method →
It is used to upload the data & then Validate the input values against given answers in the Database

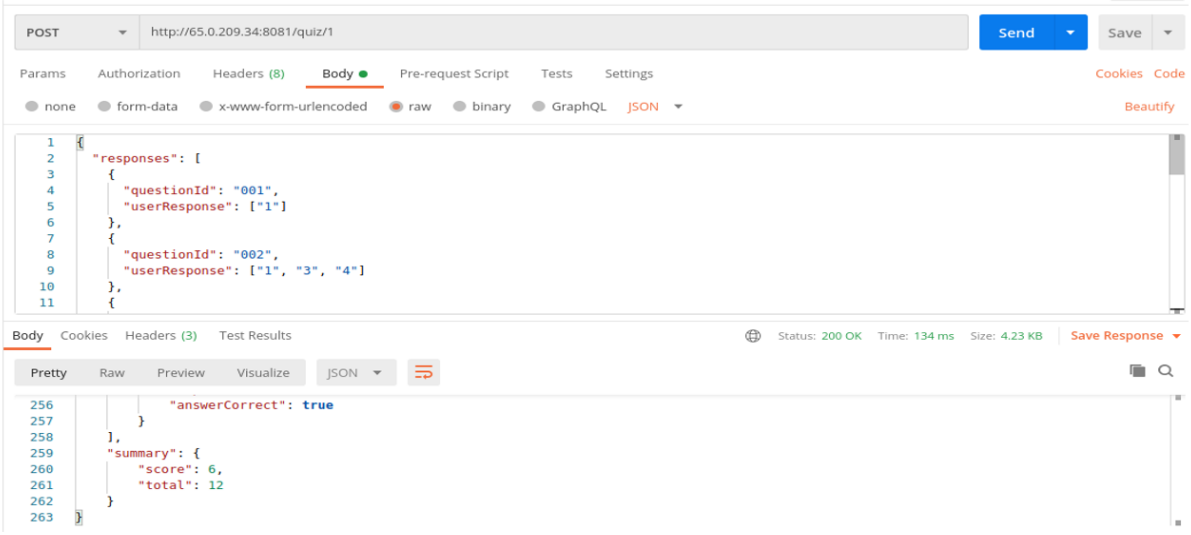

Figure 7.2: POST Method

PUT Method →
It is used to load the data in Database

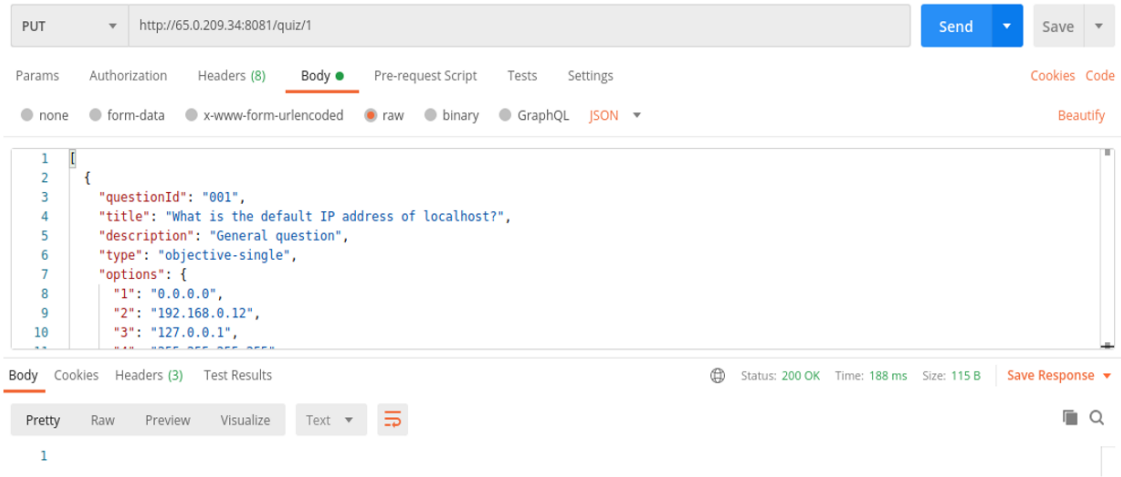

Figure 7.3: PUT Method

Chapter 8

PROJECT SUMMARY AND CONCLUSIONS

8.1 Summary

1. **Create an API to fetch quiz questions -**

   Scope of work:

   Implement GET /quiz/module-id and the corresponding request handler and response methods.

   Retrieve a list of questions from MongoDB based on the module-id.

   Skills Learned:

   Spring Boot, Spring Data, REST API, Jackson, JUnit, MongoDB

2. **Submit quiz answers and return an evaluation -**

   Scope of work:

   Accept user answers to a given quiz via a POST request

   Evaluate user answers for correctness and compute the total score

   Return a response with explanations for all answers

Skills Learned:

Scientific Debugging

3. System Testing -

Scope of work:

Unit Testing

Unit tests are typically automated tests written and run by software developers to ensure that a section of an application (known as the ”unit”) meets its design and behaves as intended.

Intended Testing

Integration testing (sometimes called integration and testing, abbreviated IT) is the phase in software testing in which individual software modules are combined and tested as a group.

Skills Learned:

JUnit, Mockito

3. Design And Technology -

Scope of work:

MVCS Architecture

Model View Controller Service or MVCS as it is popularly called, is a software design pattern for developing web applications. A Model View Controller Service pattern is made up of the following four parts

Model $\rightarrow$ The lowest level of the pattern which is responsible for maintaining data.

View $\rightarrow$ This is responsible for displaying all or a portion of the data to the user.

Controller → Software Code that controls the interactions between the Model and View.

Service → Responsible for Business Logic.

Technologies

JAVA, SpringBoot, Spring JPA.

4. DataBase -

Scope of work:

MongoDB

MongoDB is a cross-platform document-oriented database program. Classified as a NoSQL database program, MongoDB uses JSON-like documents with optional schemas.

8.2 Conclusion

Buildout is a Spring Boot Application that can accept user inputs through Check-box, Radio-buttons, Textbox and validate the values given in them against given answers.
We have Design schema and make database calls to MongoDB for CRUD operations using Spring JPA also done unit testing using JUnit and Mockito to avoid error in application and for better user experience.

Created a quiz application that supports the following API endpoint: /quiz

This endpoint supports the following operations:

GET /quiz/moduleId: fetches the quiz questions for a given module.

POST /quiz/moduleId: submits user answers and returns an evaluation.

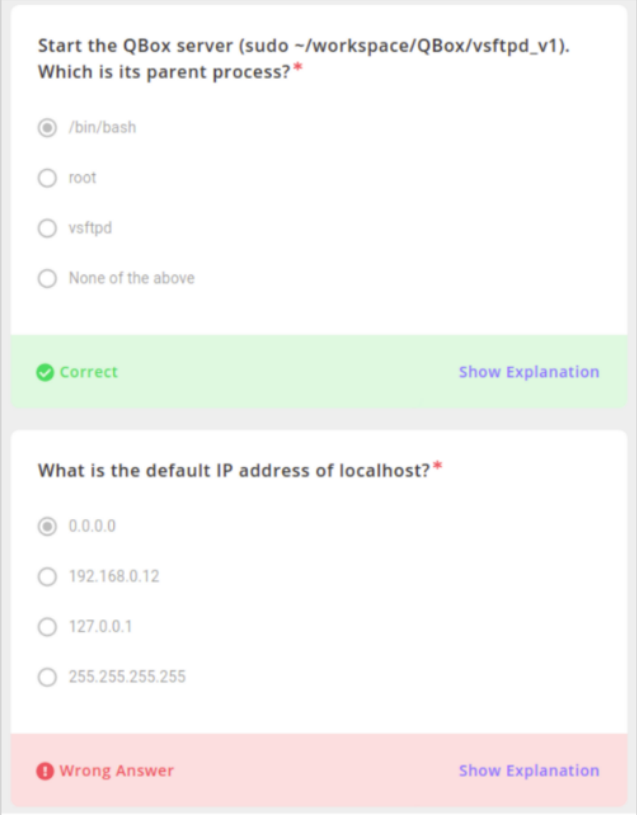

Figure 8.1: Question1

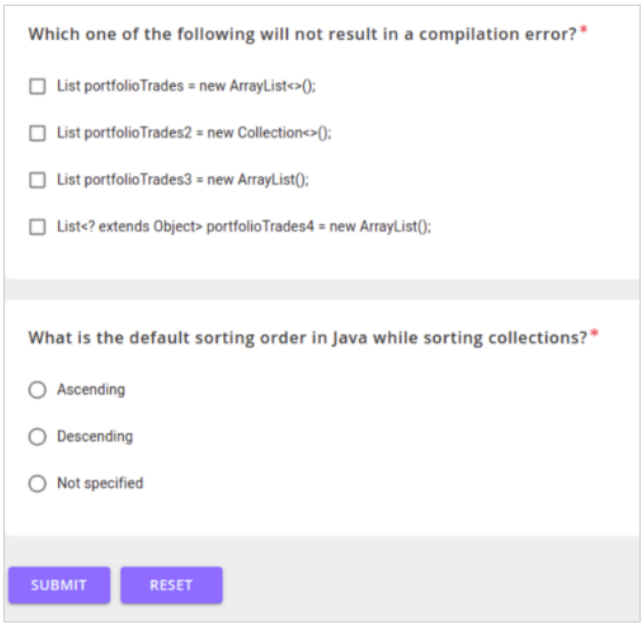

Figure 8.2: Question2

Chapter 9

FUTURE SCOPE

- Buildout is a Question and Answer form but it is more scalable because we use MVCS Architecture which helps us in providing Abstraction.
- We also use Java Interfaces so any changes in future will not effect whole Structure.
- We have used JSON which is language independent so in future we will not take care of from which client our application is used.
- We have also used Nosql database which is MongoDB so we are not focusing on specific structure of database and in future we can easily change it according to the client requirement
- This application is build on java which guarantees about security and efficiency.
- Also Created Unit Test which cover all edge cases so any update in future which can lead to error can easily detected.

References

[1] https://start.spring.io/

[2] https://www.baeldung.com/spring-boot

[3] https://www.tutorialspoint.com/spring_boot/spring_boot_unit_test_cases.htm

[4] https://medium.com/machine-words/scientific-debugging-part-1-8890b73b6c4c