



Interactive Course on ADVANCED JAVA

PROJECT GUIDE:
Mrs. N. BHUVANESWARY

NAME : PICHHAPATI VISHNU VARDHAN REDDY
REG.NO : 9919005170



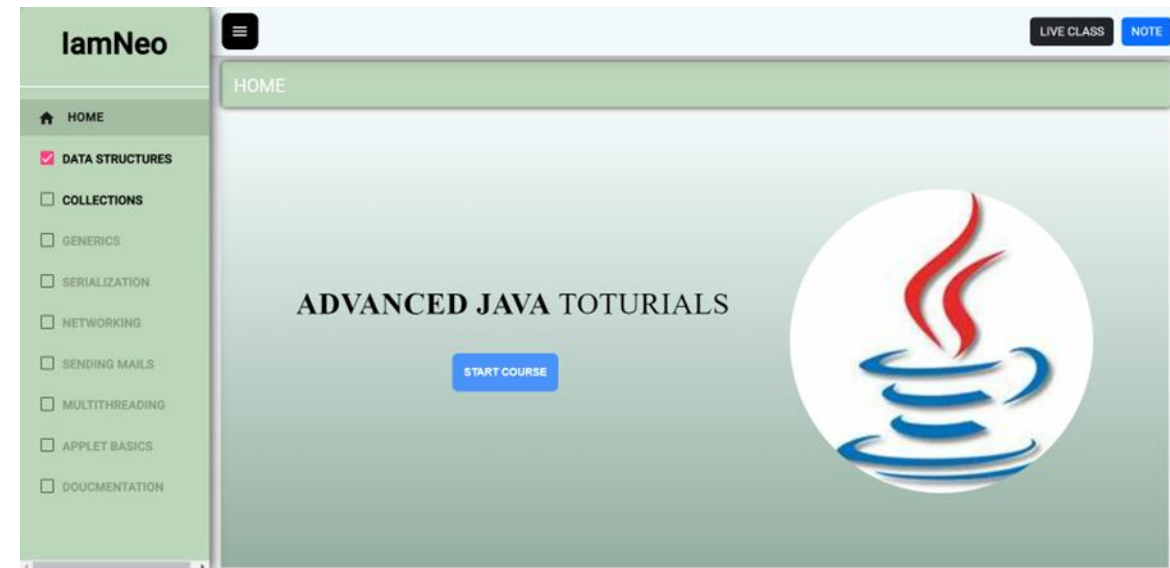
Abstract:

An interactive course on Java Advanced would typically cover advanced topics in the Java programming language, such as object-oriented programming, multithreading, networking, and more. The course would likely be designed for individuals who already have a basic understanding of Java and want to take their skills to the next level.

The course would likely be interactive, with a mix of lectures, hands-on exercises, and programming assignments. Participants may also be encouraged to work on projects in small groups or individually, and there may be opportunities for feedback and collaboration.

Objectives:

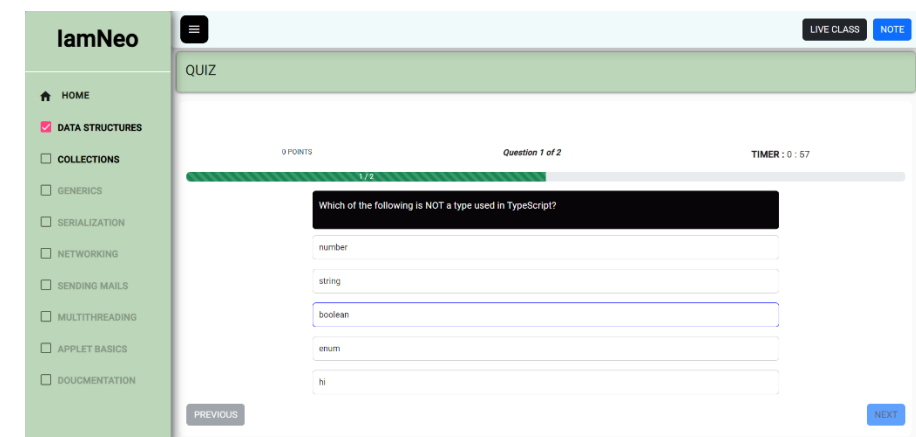
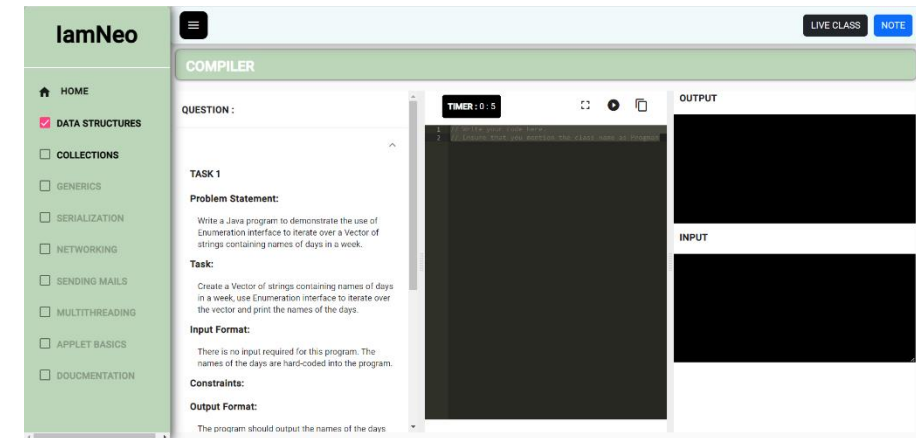
- To provide students with a strong foundation in the fundamentals of Java Advanced.
- To help students understand the core concepts of Java Advanced programming language.
- To introduce students to the most common Topics related to Java Advanced like Data structures, collections, generics, serialization, networking....etc
- To teach students how to implement and use advanced topics in Java.



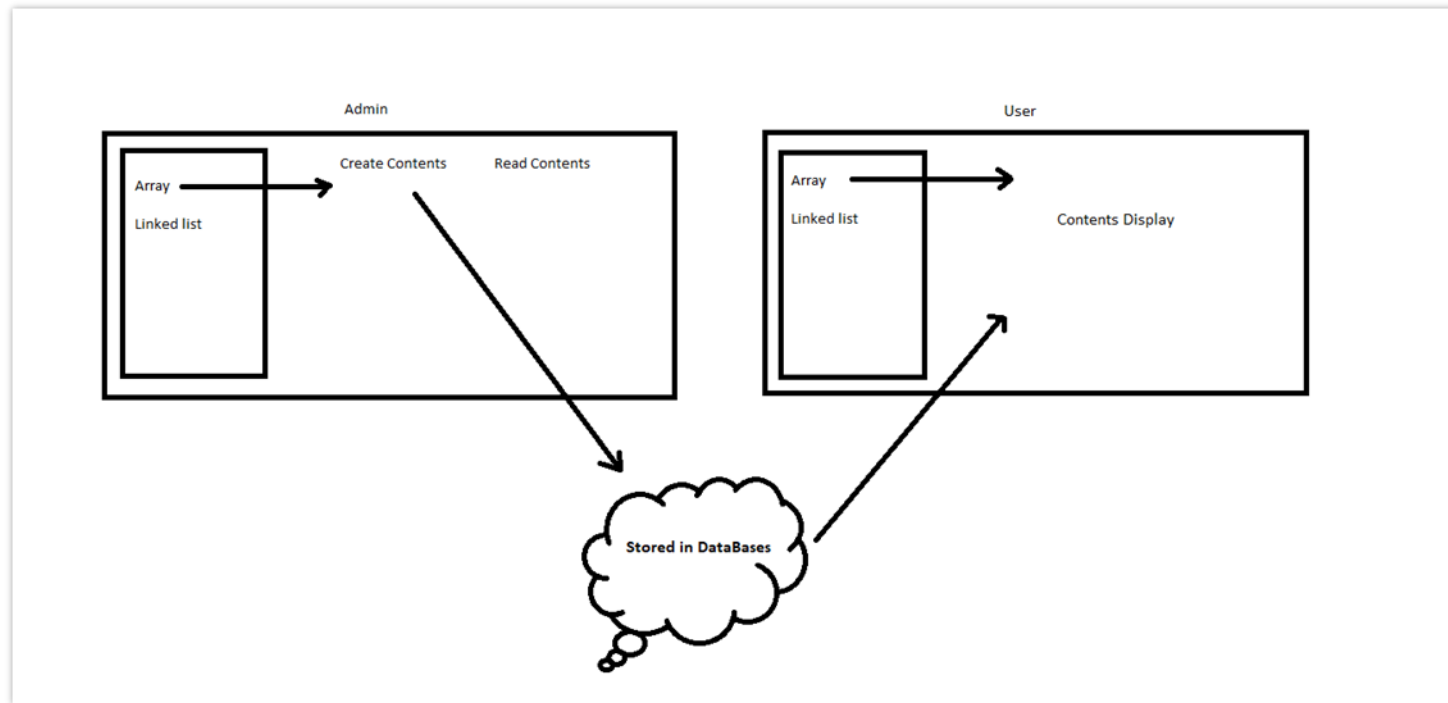
Proposed Work:

The course will consist of several modules, each focusing on a specific data structure. Each module will contain the following components

- An introduction to Java Advanced and its applications.
- An example problem, where students will apply Java to solve a real-world problem.
- Exercises and quizzes to help students practice and reinforce their learning.



Flow Chart





Front End

Angular

Bootstrap

- Bootstrap is the most popular CSS Framework for developing responsive and mobile-first websites

HTML

- it is a standard markup language used to create and design web pages.

CSS

- You can completely change the look of your website with only a few changes in CSS code.

TypeScript

- TypeScript is a strongly typed programming language that builds on JavaScript

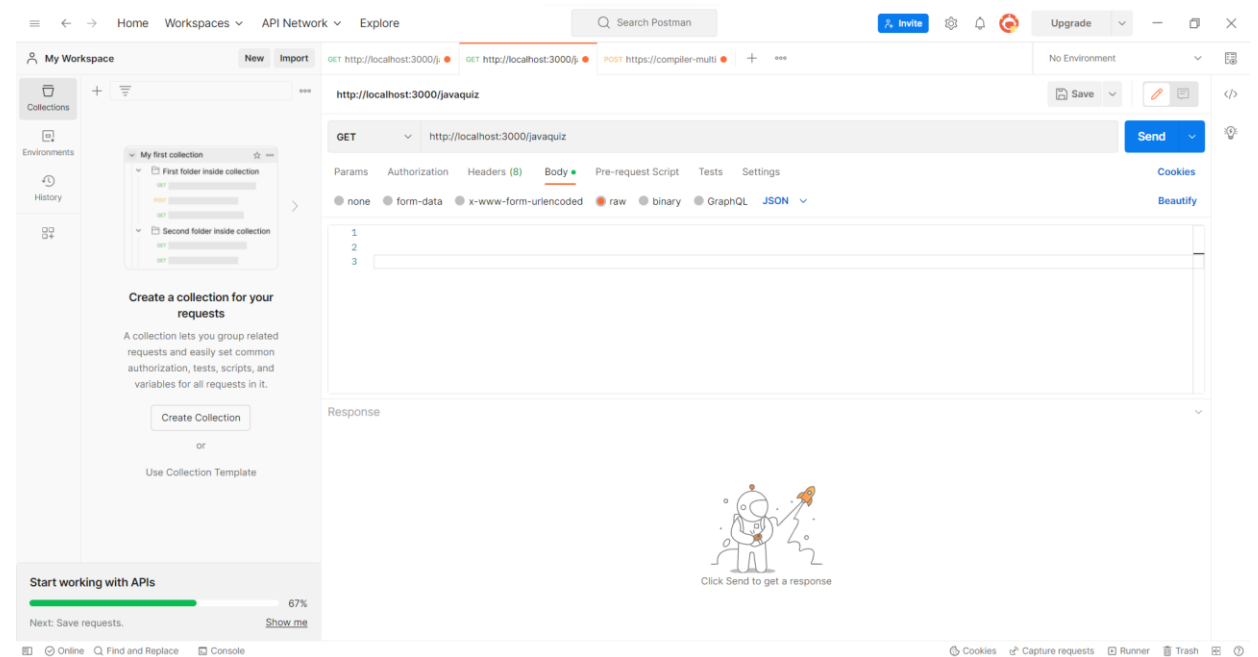
Backend End

Node.js

- Node.js is an open-source server environment
- It is free
- It runs on various platforms (Windows, Linux, Unix, Mac OS X, etc.)
- It uses JavaScript on the server

Postman (For Checking Response)

- Postman is an API platform for building and using APIs.



The ER diagram illustrates the database schema for 'Java Interview Questions'. It consists of 13 tables, each with its own set of columns and primary/foreign keys. The relationships between tables are shown using dashed lines with crow's foot notation.

Tables and their attributes:

- example_codes**: example_id BIGINT (PK), topic_id INT (FK), questions VARCHAR(1000), example_code VARCHAR(1000), output VARCHAR(1000).
- languages**: language_id INT (PK), language_name VARCHAR(1000).
- images**: image_id BIGINT (PK), topic_id INT (FK), image VARCHAR(1000).
- stack**: stack_id BIGINT (PK), topic_id INT (FK), content VARCHAR(1000).
- javacompiler**: question_id BIGINT (PK), topic_id INT (FK), id_quiz BIGINT (FK), video VARCHAR(1000), heading VARCHAR(1000), questions VARCHAR(1000), task VARCHAR(1000), inputFormat VARCHAR(1000), OutputFormat VARCHAR(1000), input VARCHAR(1000), output VARCHAR(1000), explanation VARCHAR(1000), hint VARCHAR(1000).
- datastructures_content**: datastructures_id BIGINT (PK), topic_id INT (FK), content VARCHAR(1000).
- video**: video_id BIGINT (PK), topic_id INT (FK), video VARCHAR(1000).
- enumeration**: enumeration_id BIGINT (PK), topic_id INT (FK), content VARCHAR(1000).
- interview_questions**: interview_id BIGINT (PK), topic_id INT (FK), question VARCHAR(1000), answer VARCHAR(1000).
- quiz_questions**: question_id BIGINT (PK), topic_id INT (FK), questions VARCHAR(1000).
- bitset**: bitset_id BIGINT (PK), topic_id INT (FK), content VARCHAR(1000).
- java**: topic_id INT (FK), topics VARCHAR(1000), language_id INT (FK), checkbox TINYINT(1), unable TINYINT(1).
- quiz_options**: option_id BIGINT (PK), question_id BIGINT (FK), question_options VARCHAR(1000), answer TINYINT(1).
- javaquiz**: quiz_id INT (PK), topic_id INT (FK), questions VARCHAR(1000), option1 VARCHAR(1000), option2 VARCHAR(1000), option3 VARCHAR(1000), option4 VARCHAR(1000).
- notes**: note_id BIGINT (PK), note VARCHAR(1000).
- hashtable**: hashtable_id BIGINT (PK), topic_id INT (FK), content VARCHAR(1000).

Relationships:

- example_codes** to **languages**: 1:M (example_id to language_id)
- images** to **languages**: 1:M (image_id to language_id)
- stack** to **languages**: 1:M (stack_id to language_id)
- javacompiler** to **languages**: 1:M (question_id to language_id)
- datastructures_content** to **languages**: 1:M (datastructures_id to language_id)
- video** to **languages**: 1:M (video_id to language_id)
- enumeration** to **languages**: 1:M (enumeration_id to language_id)
- interview_questions** to **languages**: 1:M (interview_id to language_id)
- quiz_questions** to **languages**: 1:M (question_id to language_id)
- bitset** to **languages**: 1:M (bitset_id to language_id)
- java** to **languages**: 1:M (topic_id to language_id)
- quiz_options** to **interview_questions**: 1:M (option_id to interview_id)
- javaquiz** to **interview_questions**: 1:M (quiz_id to interview_id)
- notes** to **interview_questions**: 1:M (note_id to interview_id)
- hashtable** to **interview_questions**: 1:M (hashtable_id to interview_id)
- example_codes** to **datastructures_content**: 1:M (example_id to datastructures_id)
- images** to **video**: 1:M (image_id to video_id)
- stack** to **video**: 1:M (stack_id to video_id)
- javacompiler** to **video**: 1:M (question_id to video_id)
- datastructures_content** to **video**: 1:M (datastructures_id to video_id)
- enumeration** to **video**: 1:M (enumeration_id to video_id)
- interview_questions** to **quiz_questions**: 1:M (interview_id to question_id)
- bitset** to **java**: 1:M (bitset_id to topic_id)
- java** to **quiz_options**: 1:M (topic_id to option_id)
- java** to **javaquiz**: 1:M (topic_id to quiz_id)
- java** to **notes**: 1:M (topic_id to note_id)
- java** to **hashtable**: 1:M (topic_id to hashtable_id)



Conclusion

In conclusion, an interactive course on Java Advanced is an excellent way to equip students with the knowledge and skills they need to succeed in the field of computer science. The objectives of the course are to provide students with a strong foundation in the fundamentals of Java Advanced, help them understand the core concepts of the Java Advanced programming language, and introduce them to the most Advanced data structures.