**COMSATS University Islamabad,   
Abbottabad Campus**

**SOFTWARE DESIGN DESCRIPTION   
(SDD DOCUMENT)**

**for**

**Interview Preparation Application**  
Version 1.0

***By***

**Hozefa Hassan Rizvi CIIT/FA20-BSE-019/ATD**

**Muhammad Hammad CIIT/ FA20-BSE-031/ATD**

**Ashar Ali CIIT/ FA20-BSE-158/ATD**

***Supervisor*Mam Sana Malik**

***Bachelor of Science in Computer Science (2020-2024)***

**Table of Contents**

**Revision History 3**

**1.** **Introduction 5**

**2.** **Design Methodology and software process model 5**

**3.** **System Overview 5**

3.1 Architectural Design 5

3.2 Process Flow/Representation 5

**4.** **Design Models [along with descriptions] 5**

**5.** **Data Design 6**

5.1 Data Dictionary 6

**6.** **Algorithm & Implementation 6**

**7.** **Software Requirements Traceability Matrix 6**

**8.** **Human Interface Design 7**

8.1 Screen Images 7

8.2 Screen Objects and Actions 7

**9.** **Appendix I 7**

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason for changes** | **Version** |
|  |  |  |  |
|  |  |  |  |

**Application Evaluation History**

|  |  |
| --- | --- |
| **Comments (by committee)**  **\*include the ones given at scope time both in doc and presentation** | **Action Taken** |
|  |  |
|  |  |

**Supervised by**

**Mam Sana Malik**

Signature\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Introduction**

The project has focused on developing an **"Interview Preparation Application,"** which aims to assist software engineering graduates in excelling in their job interviews. The project encompasses various modules, including **user authentication and profile setup**, a comprehensive **question bank**, **community engagement features**, **expert interaction capabilities**, **mock interview**, and **administrative management functionalities**. These modules work cohesively to create an integrated platform that aids in interview preparation and skill enhancement.

**Design methodology and software process model**

**Design methodology:**

For the design methodology, the project has adopted a procedural approach to accommodate the specific requirements of the chosen technology stack. React Native, which utilizes JavaScript, for the most part operates with a procedural design approach, which emphasizes step-by-step execution and the use of functions for streamlined development. Although React Native incorporates certain object-oriented concepts, the project primarily aligns with a procedural design to ensure compatibility and optimal performance within the chosen framework.

**Software Process Model:**

Regarding the software process model, the project has followed the practices of the Agile methodology. Agile has been selected to facilitate iterative development, enhance flexibility, and allow for continuous user feedback. Given the dynamic nature of the project requirements and the necessity for adaptability in the development process, Agile stands out as the most suitable software process model. The emphasis on collaboration, adaptability, and user-centered development inherent in the Agile methodology has contributed significantly to the project's progress and success.

**System overview**

**Purpose**

The **Interview Preparation Application** aims to provide comprehensive support to software engineering graduates by facilitating effective interview preparation and skill enhancement.

**Functionality:**

* **Question Bank:** The application houses an extensive database of industry-specific interview questions and answers, allowing users to prepare for a diverse range of potential interview scenarios.
* **Expert Interaction:** Users can connect with seasoned industry professionals to seek guidance and receive practical insights into effective interview strategies, communication techniques, and industry trends.
* **Mock Interview Analysis:** The application offers a sophisticated mock interview feature that provides real-time feedback on users' non-verbal communication, practical skills, and confidence levels, helping them identify areas for improvement.
* **Community Engagement:** Encouraging a collaborative environment, the application facilitates active participation within the software engineering community, fostering discussions, and the sharing of valuable interview experiences and tips among peers.
* **User Authentication and Profile Setup:** Users can easily create and manage their profiles, providing essential information related to their technical interests, academic background, and skillset, optimizing their interaction with the application's features.

**Context:**

Designed within the context of the fast-paced and competitive software engineering industry, the application emphasizes the importance of not only technical proficiency but also effective communication and practical skills during interviews.

**Design:**

With a user-centric and intuitive design, the application ensures a seamless and engaging experience for users, encouraging active participation and collaboration within the software engineering community.

**Background**

The development of the Interview Preparation Application is driven by the need to bridge the gap between theoretical knowledge and the practical skills required for successful career placement in the software engineering domain.

**Architectural design**

For the Interview Preparation Application, a suitable architecture would be the Model-View-Controller (MVC) architecture. This choice is primarily based on the need for a structured and organized approach to handle the application's complex functionalities. MVC provides clear separation between the application's data, user interface, and business logic, promoting better code maintainability, scalability, and reusability.

**Diagram:**

A diagram of a process

Description automatically generated

**From the above diagram:**

**Model:**

The model represents the data and business logic of the application. In this case, the model would include things like:

* User profiles
* Interview questions and answers
* Expert profiles
* Mock interview results

**View:**

The view is the user interface of the application. It is responsible for displaying the data from the model to the user and taking input from the user. In this case, the view would include things like:

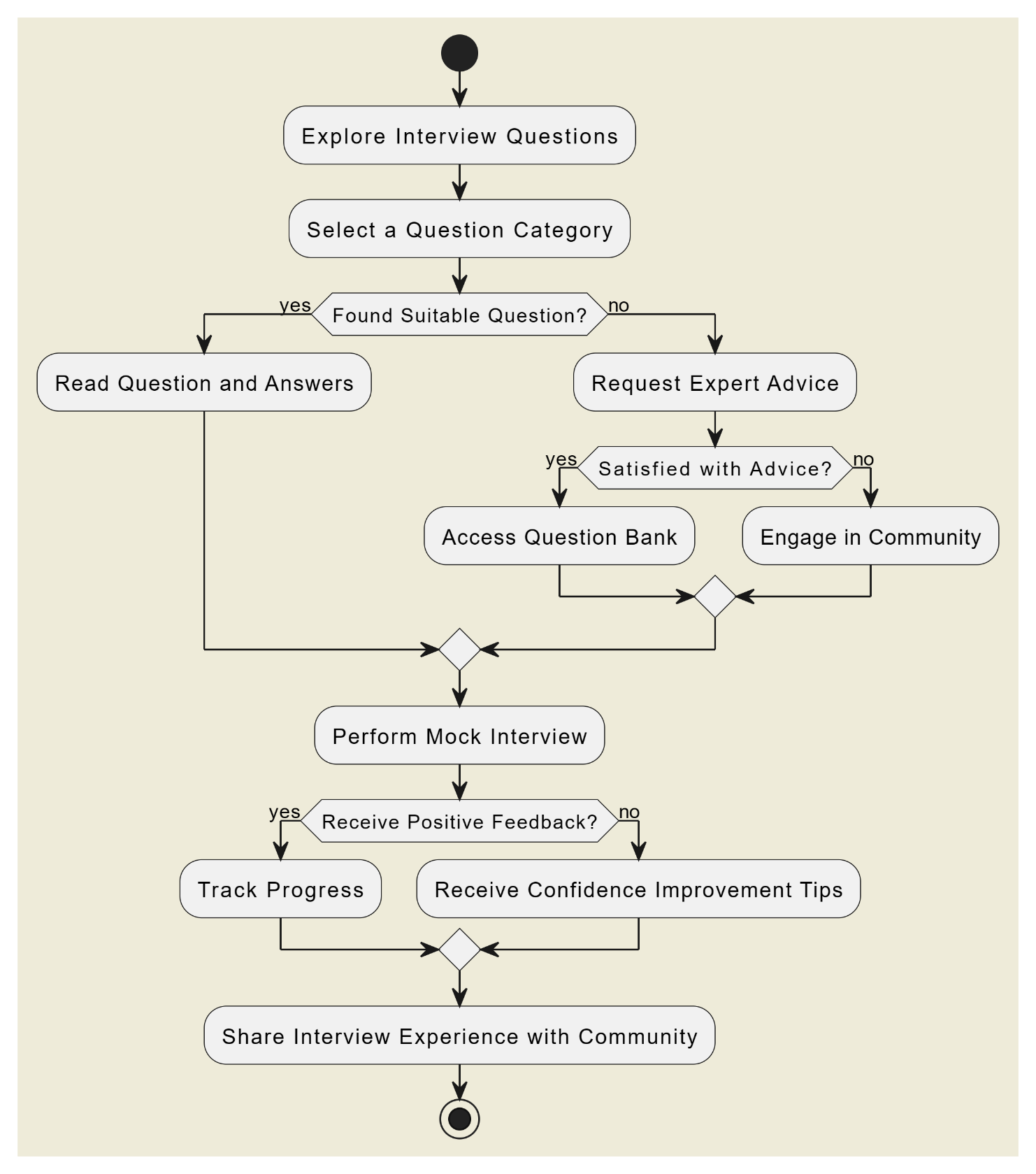
* Home page with a list of subfields related to software engineering.
* Question Bank page with a list of interview questions
* Expert Panel page with a list of experts
* Mock Interview page with an interface for the user to practice answering interview questions.
* Community engagement where users can ask questions and share insights.

**Controller:**

The controller acts as an intermediary between the model and the view. It is responsible for handling user requests, updating the model, and returning the updated view to the user. In this case, the controller would handle things like:

* Retrieving a list of interview questions for a given subfield
* Saving the user's answers to a mock interview
* Forwarding a question to the Expert Panel
* Posting a message to the discussion forum

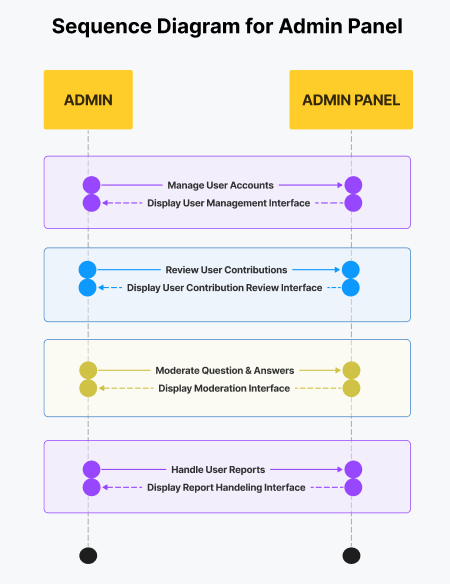
**Process flow/Representation**



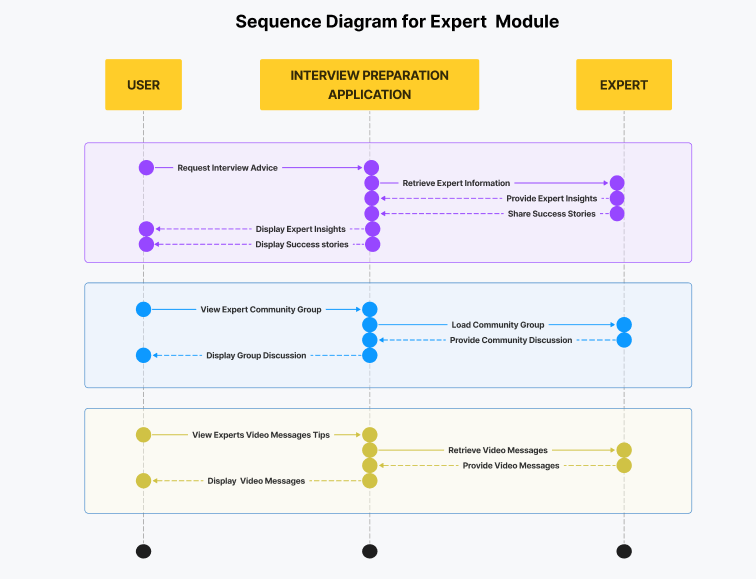
**Design models:**

**Sequence Diagrams:**

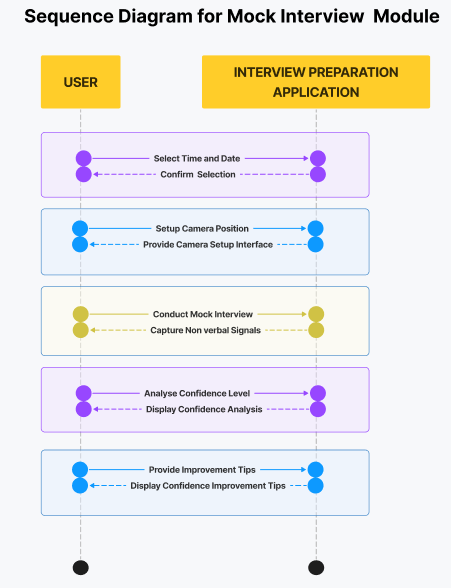
1. **Sequence Diagram (Admin Module):**



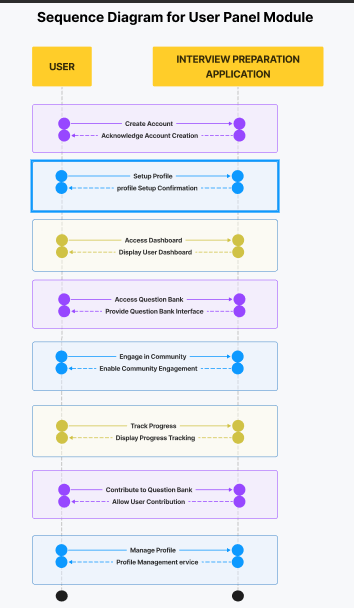
1. **Sequence Diagram (Expert Module):**



1. **Sequence Diagram (Mock Interview Module):**

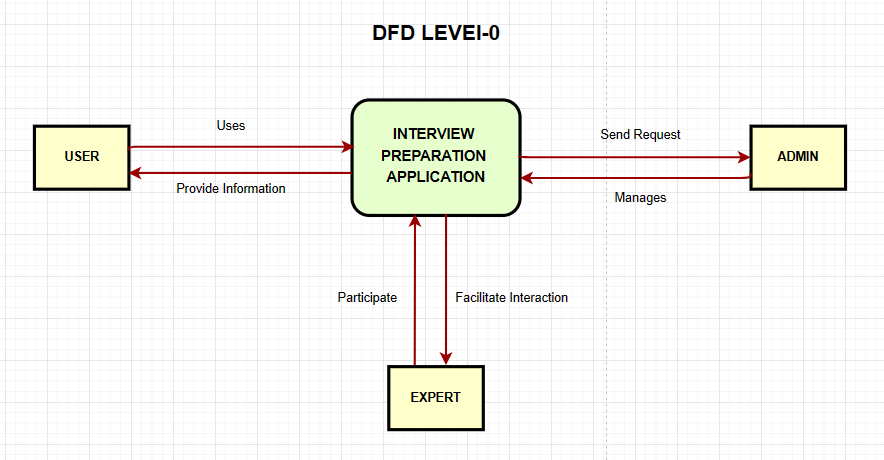


1. **Sequence Diagram (User Module):**

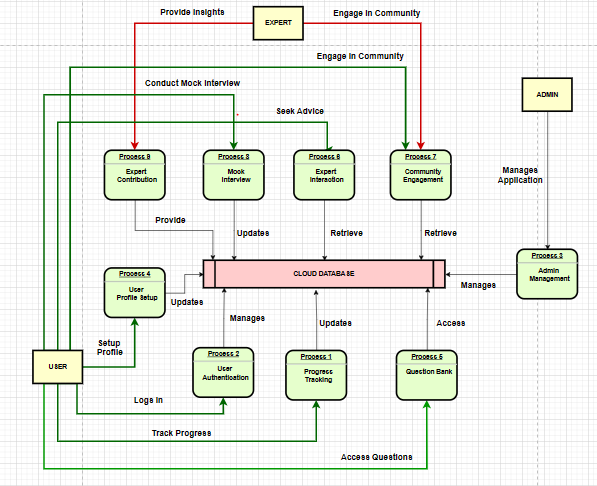


**Data Flow Diagram**

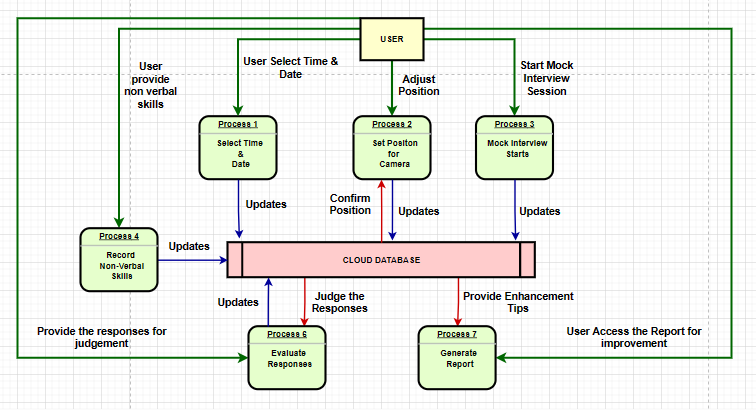
**DFD Level-0:**



**DFD Level-1:**



**DFD Level-2(Mock Interview Process):**



**State Transition Diagram:**

**Data design**

Explain how the information domain of your system is transformed into data structures. Describe how the major data or system entities are stored, processed and organized.

List any databases or data storage items.

**Data dictionary**

Alphabetically list the system entities or major data along with their types and descriptions. If you provided a functional description, list all the functions and function parameters. If you provided an OO description, list the objects and its attributes, methods and method parameters.

**Algorithm & Implementation**

In this section, we take a closer look at what each component does in a more systematic way. Provide a summary of your algorithm for each function listed in procedural description language (PDL) or pseudo code.

If you gave an OO description, summarize each object member function for all the objects listed in PDL or pseudo code. Describe any local data when necessary.

**Software requirements traceability matrix**

This section should contain a table that summarizes how each software requirement has been met in this document. The tabular format permits one-to-one and one-to-many relationships to be shown.

**Table 1 Requirements Traceability Matrix**

|  |  |  |  |
| --- | --- | --- | --- |
| **Req. Number** | **Ref. Item** | **Design Component** | **Component Items** |
| FR01 | Class Diagram | ClassName | FunctionName(s) |
| OR | | | |
| FR01 | DFD | DiagramNumber/Level | FunctionName(s) |

**Human interface design**

Describe the functionality of the system from the user’s perspective. Explain how the user will be able  to use  your system to complete  all the  expected  features and  the  feedback  information that will be displayed for the user.

**Screen images**

Display screenshots showing the interface from the user’s perspective. These can be hand-drawn, or you can use an automated drawing tool. Just make them as accurate as possible. (Graph paper works well.)

**8.2 Screen objects and actions**

A discussion of screen objects and actions associated with those objects

**Appendix I**

* How to design using UML (OOP): For guidance please follow the instructions mentioned in the link: http://agilemodeling.com/artifacts/
* How and when to design ER diagrams: For guidance please follow the instructions mentioned in the link:

<http://people.inf.elte.hu/nikovits/DB2/Ullman_The_Complete_Book.pdf>

* Data flow diagrams: For guidance please follow the instructions mentioned in the link and book:
  + http://www.agilemodeling.com/artifacts/dataFlowDiagram.htm
  + Software Engineering –A Practitioner’s approach by Roger Pressman
* Architecture diagram: For guidance please follow the instructions mentioned in the link and book:
  + Ian Sommerville – Software Engineering 9th Edition– Chapter 6