**A Synopsis**

**On**

**“DebugDesk – Collaborate, Code, Solve – Together in Real-Time”**

Submitted in partial fulfillment of the requirements for the award of the Degree of

**Bachelor of Computer Application**

**of**

**Poornima University, Jaipur**

**Session: 2022-2025**

****

**Submitted By:**

**Milan Sharma [2022PUFCEBCYX10941]**

**Ketan Kedar [2022PUFCEBCYX12161]**

**III Year, CYBER SECURITY**

**Submitted to:** Department of Computer Application

**Faculty of Computer Science & Engineering,**

**Poornima University** Ramchandrapura, Sitapura Ext., Jaipur, Rajasthan

1. Abstract:

The Debug Desk is a cutting-edge Q&A platform designed to cater specifically to the developer community. With features like advanced markdown support, smart search, and intelligent tagging, Debug Desk simplifies the process of finding solutions to complex technical issues. By integrating a clean and modern user interface with a robust backend powered by Next.js and Appwrite, this platform ensures a seamless and responsive experience across all devices. Debug Desk aspires to be a one-stop solution for developer collaboration, learning, and knowledge sharing.

1. Introduction:

Debug Desk was conceptualized as a response to the growing need for a developer-centric knowledge-sharing platform. Traditional Q&A sites often fall short in addressing the specific requirements of developers, such as efficient code formatting, real-time collaboration, and intuitive search tailored to programming contexts.

The platform leverages a state-of-the-art tech stack, including Next.js for high-performance web applications, Appwrite for secure authentication and real-time data synchronization, and Tailwind CSS for responsive and clean designs. Additionally, the reputation system incentivizes quality contributions, fostering a thriving and supportive developer community.

**Rationale**

With the rise of remote work and distributed development teams, there is a growing demand for platforms that facilitate seamless communication and problem-solving. Debug Desk fills this gap by creating a dedicated space where developers can focus on sharing knowledge without distractions, ultimately contributing to the global tech community's growth and efficiency.

1. Problem Statement:

Traditional Developers often face hurdles in finding precise, reliable, and community-verified solutions to their technical challenges. Existing platforms provide generalist solutions that lack tailored functionalities such as syntax-aware search engines, developer-oriented tags, and markdown support with syntax highlighting. Debug Desk seeks to bridge these gaps by creating a platform designed explicitly for developers, addressing the pain points of traditional knowledge-sharing tools.

1. Objectives & Scope:

1. **Enable efficient knowledge sharing** among developers using advanced markdown and real-time collaboration tools.

2. **Provide a structured and intelligent search experience** with context-awareness for technical queries and error patterns.

3. **Promote a culture of quality contributions** through a reputation system that rewards helpful answers and solutions.

4. **Facilitate personalized experiences** with custom dashboards and user profiles tailored to individual developers’ expertise and interests.

5. **Expand developer capabilities** with future integrations such as IDE plugins and AI-powered answer suggestions.

The project aims to serve professional developers, programming enthusiasts, and students, creating a knowledge-sharing ecosystem that fosters continuous learning and growth.

Literature Review:

Several platforms, such as Stack Overflow and GitHub, serve developers but lack a cohesive blend of features tailored to knowledge-sharing. Stack Overflow focuses on question-and-answer formats but lacks personalized dashboards and interactive collaboration tools. GitHub supports code versioning and collaboration but is not optimized for Q&A purposes. Debug Desk combines the best aspects of these platforms while introducing unique features like smart search, intelligent tagging, and advanced markdown capabilities. Additionally, it aligns with modern development trends like responsive design and real-time updates, addressing shortcomings in current solutions.

1. Feasibility Study:

Technical Feasibility:

The proposed technologies are mature and well-supported, ensuring efficient development and deployment. Next.js allows for high-performance server-side rendering and static site generation, while Appwrite simplifies user authentication and data management.

Economic Feasibility:

By using open-source libraries and frameworks like Zustand and Tailwind CSS, development costs are minimized. The platform’s scalable architecture also ensures cost efficiency during future expansions.

Operational Feasibility:

The intuitive interface, responsive design, and dark mode support make Debug Desk accessible and appealing to a wide audience, ensuring user adoption and satisfaction.

Social Feasibility:

By promoting collaboration and knowledge sharing, Debug Desk has the potential to significantly benefit the developer community and contribute to faster problem-solving and innovation.

1. Modules in Proposed Project:

1. User Authentication Module:

* Features secure login and sign-up processes using Appwrite
* Includes support for third-party authentication (e.g., Google, GitHub)..

2. Q&A Module:

* Enables users to ask and answer questions with advanced markdown and code formatting
* Incorporates real-time updates for collaborative discussions.

3. Search Module:

* Implements a context-aware search engine for fast and accurate query results.
* Includes filters for tags, users, and topics.

4. Reputation & Tagging System:

* Incentivizes user engagement through a reputation score based on community votes.
* Provides intelligent tagging suggestions to enhance content discoverability.

5. Personalized Dashboard:

* Tracks user activity, reputation, and impact metrics.
* Displays contributions, achievements, and frequently visited tags/topics.

1. Data Flow Diagram (DFD):

A diagram of a user data flow

Description automatically generatedThe DFD illustrates how data flows between users, the server, and the platform's features, emphasizing interactions like login, resource access, and collaboration.

A diagram of a data flow

Description automatically generatedA diagram of a data flow

Description automatically generated

A diagram of a process

Description automatically generated

ER Diagram DebugDesk
8.Entity-Relationship (ER) Diagram:

Explanation of the Diagram:

* Questions: Stores details of questions with fields like id, title, content, tags, attachmentId, and authorId. Each question can have attachments, comments, answers, and votes.
* Answers: Represents answers to questions. Each answer is linked to a question (questionId) and has an authorId.
* Comments: Stores comments linked to either a question or an answer via type (enum) and typeId. Each comment has an authorId.
* Votes: Tracks votes (upvote/downvote) for questions, answers, or comments via type (enum) and typeId. It includes votedById (user who voted) and voteStatus (enum for upvote/downvote).
* Users: Stores user data, including id, email, name, profilePhoto, and phoneNumber.
* Question Attachments: Manages file attachments for questions, identified by id and filetype

9.Technical Details:

**Hardware Requirements:**

**Server**: Cloud-based hosting (e.g., AWS, Google Cloud, or Vercel).

**User Devices**: Smartphones, tablets, or PCs with internet access.

**Software Requirements:**

**Backend**: Node.js (integrated with WebSocket), Next.js, Appwrite.

**Frontend**: React, Tailwind CSS.

**Database**: Appwrite’s built-in document-based database or MongoDB.

**Platform:** Web.

10.Methodology:

**Approach**: Agile methodology to deliver iterative and incremental features.

**Techniques**: Component-based development, RESTful API design

**Tools**: Visual Studio Code, Postman, Git, and CI/CD pipelines.

Steps:

1. Requirement Analysis
2. Design Prototyping
3. Development of Core Modules
4. Integration of Real-Time Features (WebSocket)
5. Testing and Debugging
6. Deployment and Maintenance

11.Proposed WorkFlow:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Proposed Phase and Duration Chart** | | | | |
| **Phase** | **Description** | **Duration (Weeks)** | **Start Week** | **End Week** |
| Phase 1: Requirement Analysis | Define project scope, goals, and gather requirements. | 2 | Week 1 | Week2 |
| Phase 2: Literature Review | |  | | --- | |  |  |  | | --- | | Conduct research on relevant literature and existing systems. | | 2 | Week3 | Week4 |
| Phase 3: Feasibility Study | Analyze technical, operational, and financial feasibility. | 2 | Week5 | Week6 |
| Phase 4: Design | Create system architecture, UI/UX designs, and database. | 2 | Week 7 | Week 9 |
| Phase 5: Development | Implement the project (coding, integration, etc.). | 2 | Week 10 | Week 15 |
| Phase 6: Testing | Perform functional, integration, and user acceptance testing. | 2 | Week 16 | Week 17 |
| Phase 7: Documentation & Deployment | Complete system documentation and deploy. | 2 | Week 18 | Week 19 |

Gantt Chart:

+--------------------------------------------------------------+

| Phase | Duration (Weeks) |

+--------------------------------- +-------------------------+

| Phase1: Requirement Analysis | ■■■■ |

| | Week 1 to Week 2 |

+----------------------------------- +------------------------+

| Phase 2: Literature Review | ■■■■ |

| | Week 3 to Week 4 |

+------------------------------------+------------------------+

| Phase 3: Feasibility Study | ■■■■ |

| | Week 5 to Week 6 |

+----------------------------------- +-------------------------+

| Phase 4: Design | ■■■■■ |

| | Week 7 to Week 9 |

+------------------------------------+-------------------------+

| Phase 5: Development | ■■■■■■■■■■■ |

| | Week 10 to Week 14 |

+------------------------------------ +-------------------------+

| Phase 6: Testing | ■■■■|

| | Week 15 to Week 16 |

+------------------------------------ +------------------------+

| Phase 7: Documentation | ■■■■ |

| & Deployment | Week 17 to Week 18 |

+------------------------------------+-------------------------+

9.

 **Phase 1 (Requirement Analysis)** runs from **Week 1 to Week 2**.

The focus is on understanding the project scope, objectives, and gathering requirements.

 **Phase 2 (Literature Review)** is scheduled from **Week 3 to Week 4**,

involving a review of existing research, solutions, or case studies relevant to the project.

 **Phase 3 (Feasibility Study)** follows from **Week 5 to Week 6**,

where the feasibility of the project is analyzed, focusing on technical, operational,

and financial aspects.

 **Phase 4 (Design)** runs from **Week 7 to Week 9**,

where the system's architecture and design (UI/UX, database) are created.

 **Phase 5 (Development)** takes the longest, running from **Week 10 to Week**

This phase involves the actual implementation, including coding,

integration, and setting up the environment.

 **Phase 6 (Testing)** spans **Week 15 to Week 16**,

where the system undergoes functional testing, integration testing, and user acceptance testing.

 **Phase 7 (Documentation & Deployment)** takes place from **Week 17 to Week 18**,

where the final project documentation is completed, and the solution is deployed.

Pert Chart:

+---------------------------+

| Phase 1: Requirement |

| Analysis |

+--------------------------+

|

(After completion)

v

+---------------------------+

| Phase 2: Literature Review|

+---------------------------+

|

(After completion)

v

+---------------------------+

| Phase 3: Feasibility Study|

+---------------------------+

|

(After completion)

v

+---------------------------+

| Phase 4: Design |

+---------------------------+

|

(After completion)

v

+---------------------------+

| Phase 5: Development |

+---------------------------+

|

(After completion)

v

+---------------------------+

| Phase 6: Testing |

+---------------------------+

|

(After completion)

v

+---------------------------+

| Phase 7: Documentation|

| &Deployment |

+---------------------------+

1. **Phase 1: Requirement Analysis** is the starting point.

2. After **Requirement Analysis** is completed, the project moves into

**Literature Review**.

3. Upon completion of the **Literature Review**, the project advances to the

**Feasibility Study** phase.

4. After **Feasibility Study** is done, the project moves on to**Design**.

5. Once **Design** is completed, the team moves to **Development**.

6. **Testing** follows the completion of **Development**, ensuring the product works

as expected.

7. Finally, **Documentation & Deployment** is done after **Testing** to compl

the project and prepare it for deployment

**13.Conclusion:**

Debug Desk represents a significant leap forward in how developers collaborate, share knowledge, and solve technical challenges. By integrating advanced features such as real-time collaboration powered by WebSocket, context-aware search, and intelligent tagging, this platform addresses the limitations of traditional Q&A systems. Its clean, modern user interface and robust architecture ensure a seamless experience for developers, fostering a community-driven ecosystem for innovation and learning.

The project not only showcases the effective use of modern technologies like Next.js, Appwrite, and Tailwind CSS but also emphasizes scalability, security, and user engagement. With its comprehensive roadmap, including features like AI-powered suggestions and interactive code playgrounds, Debug Desk is poised to grow into a cornerstone platform for developers worldwide.

Through this project, the team has gained valuable experience in full-stack development, real-time systems, and collaborative software design, laying the foundation for future advancements in the field of software development. Debug Desk is not just a solution for today's challenges; it is a platform designed to evolve and adapt, meeting the ever-changing needs of the developer community.

14.Team Members:

**Student-1**:-



Name: Milan Sharma

Registration No: 2022PUFCEBCYX10941

Role in Project: Leader

Signature:

**Student-2** :-



Name: Ketan Kedar

Registration No: 2022PUFCEBCYX12161

Role in Project: Member

Signature:

15. Guide Name:MR. Tesheen Ahmad Bhat

**Designation:** Assistant Professor

**Signature:**