DESIGN AND DEVELOPMENT OF A WEB BASED COURSE MATERIAL MANAGEMENT SYSTEM FOR ENGINEERING STUDENTS IN THE UNIVERSITY OF BENIN

OKOYOMOH BLESSING OGHENESENI

ENG 20006326

DEPARTMENT OF INDUSTRIAL ENGINEERING

FACULTY OF ENGINEERING

UNIVERSITY OF BENIN

BENIN CITY

NOVEMBER 2025

DESIGN AND DEVELOPMENT OF A WEB BASED COURSE MATERIAL MANAGEMENT SYSTEM FOR ENGINEERING STUDENTS IN THE UNIVERSITY OF BENIN

OKOYOMOH BLESSING OGHENESENI

ENG 20006326

SUPERVISED BY DR. O. O. ASHAMA

A PROJECT SUBMITTED TO THE DEPARTMENT OF INDUSTRIAL ENGINEERING, FACULTY OF ENGINEERING, UNIVERSITY OF BENIN, BENIN CITY

IN PARTIAL FULFILMENT OF THE REQUIREMENT FOR THE AWARD OF BACHELOR OF ENGINEERING (B.ENG) DEGREE IN INDUSTRIAL ENGINEERING

NOVEMBER 2025

CERTIFICATION

This is to certify that this research work on the "Design and development of a web based course material management system for engineering students" was carried out by ENG2006326 of the Department of Industrial Engineering, University of Benin, Benin City.

---------- ----------

DR. O. O. ASHAMA Date

Supervisor

---------- ----------

ENGR. DR. (MRS) I. C. ILLUOBE Date

Project Co-ordinator

---------- ----------

PROF. PATRICK E. AIMOLEHMEN Date

HOD

DEDICATION

I dedicate this work first and foremost to God Almighty, whose mercy and grace enabled me to begin and successfully complete this project. His divine guidance, strength, and wisdom saw me through every stage of this academic journey.

ACKNOWLEDGEMENT

I am deeply grateful to my parents, Pastor Bayode and Mrs. Betty Okoyomoh, for their unending love, prayers, and unwavering support throughout my academic journey.

My heartfelt appreciation also goes to Late Barr. Akeens Ade-Akhani, Aunty Praise, Aunty Evelyn, and every member of my family for their constant encouragement and care during my stay in school I do not take it for granted.

My profound gratitude is extended to my project supervisor, Dr. O. O. Ashama, for her continuous support, encouragement, and insightful guidance. Her advice and constructive feedback were instrumental in shaping this project to its successful completion.

To my elder brother, Dr. Victory Okoyomoh, who has always been there for me throughout my studies, and to my siblings Peace, David, and Zeme, I say a sincere thank you.

A special note of appreciation goes to Ogheneseni, for always pushing yourself further in this academic race, and to Ochuwa, for a wholesome and supportive friendship.

Finally, I would like to thank my friends, course mates, and colleagues (The Boys, Albert, Bobby, Alex, Stephanie, Joshua, Laurel)

To my lecturers Dr. N. H. Osadiaye, Dr. Ikpoza, and Dr. Tina for their encouragement, collaboration, and contributions to my academic journey. Thank you all very much

ABSTRACT

This project aimed to design and implement UnibenEngVault, a web-based platform created to provide engineering students of the University of Benin with centralized and convenient access to academic resources such as lecture notes, past questions, and tutorial materials. The initiative was motivated by the persistent difficulty students face in sourcing relevant study materials due to the lack of a unified and structured digital repository within the Faculty of Engineering.

The development followed a structured methodology consisting of project planning and approval, problem analysis, system design, development, and deployment. The UI/UX design was created using Figma to ensure an intuitive and visually appealing interface. The frontend was developed with ReactJS, while the backend was implemented using Python (Flask) and PostgreSQL for database management. AWS S3 was utilized for cloud storage, and deployment was achieved through Docker, GitHub Actions (CI/CD), and DigitalOcean Droplets to ensure scalability and reliability. Security measures such as session authentication, CORS configuration, and bcrypt password hashing were implemented, alongside thorough unit, integration, and user testing.

The final outcome is a secure, functional, and user-friendly platform that enables students to easily access department- and level-specific academic materials. UnibenEngVault enhances academic collaboration, improves resource accessibility, and provides a sustainable technological solution to the long-standing challenge of academic material distribution within the Faculty of Engineering.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER ONE

INTRODUCTION

1.1 Background to the study

The Faculty of Engineering at UNIBEN is one of the most academically demanding faculties, with students required to access a wide range of course materials, lecture notes, past questions, project resources, and research articles across diverse technical fields. However, there is currently no centralized digital system where these materials are stored, updated, or shared efficiently. As a result, students often depend on fragmented sources such as WhatsApp groups, flash drives, outdated handouts, and personal contacts with course representatives or alumni to obtain critical learning materials. This informal and unreliable method of information dissemination not only limits equal access but also hampers academic performance, collaboration, and research continuity.

Given the pressing need for a reliable, scalable, and user-friendly platform, this study proposes the design and development of a web-based course material management system specifically for engineering students at UNIBEN. The system will serve as a centralized repository where lecturers can upload and update course materials, and students can access them anytime improving resource sharing, academic organization, and digital preparedness.

This project not only addresses a critical local challenge but also contributes to the broader movement of educational innovation in Nigerian tertiary institutions. It demonstrates how student-led, faculty-driven solutions can enhance the learning experience, reduce academic gaps, and lay the groundwork for long-term digital transformation in higher education.

1.2 Statement of the problem

In an age where technology drives innovation and shapes the future of education globally, Nigeria's educational system remains heavily under-integrated with modern technological tools. This deficiency is particularly evident in the Faculty of Engineering, where the gap between expected technological proficiency and the realities of educational delivery continues to widen.

Despite being a faculty traditionally associated with innovation and advancement, many engineering departments across Nigerian universities still operate with minimal digital infrastructure. Students are often reliant on outdated textbooks, limited access to scientific journals, and underfunded libraries, while opportunities to engage with virtual laboratories, design software, and online learning platforms remain scarce.

This lack of access to adequate and up-to-date learning materials has significant implications. It limits students' ability to conduct independent research, weakens their practical exposure, and restricts their competitiveness both locally and globally. Even more concerning is that these issues persist in environments where the technical expertise to solve them through local, engineering-led innovation already exists but is underutilized.

This project, therefore, seeks to address this problem by implementing a technology-based solution within the Faculty of Engineering, targeting the inadequate access to educational materials as a case study. By leveraging low-cost, scalable technology and engineering principles, this project aims to demonstrate how in-house innovations can bridge the resource gap and serve as a model for broader adoption across the Nigerian tertiary education system.

1.3 Aims and Objectives

The study is guided by the following objectives:

1. Investigate the current level of access to educational materials and digital tools within the Faculty of Engineering.
2. Analyze data gotten from investigation and document all information gotten.
3. Collect, digitize and organize lecture materials, textbooks, and practical resources.
4. Conduct user story research and create user flow chart for the web app.
5. Design and develop a web app that will be used to access the course materials.
6. Deploy and test the app in all departments in the faculty of engineering.
7. Assess the effectiveness of the system in improving access to academic resources and overall student learning outcomes.

1.4 Significance of study

This study holds substantial significance, not only for the Faculty of Engineering but for the broader educational and technological landscape in Nigeria. Its potential impact spans academic, institutional, national, and social dimensions:

1 Academic Development

Engineering education depends heavily on access to accurate, current, and practical learning resources. In Nigeria, limited access to such materials has historically hindered effective learning and innovation. This project addresses this gap by providing students with digital access to core engineering textbooks, lecture notes, design manuals, and tutorial content, thereby supporting deep and independent learning. The availability of offline-accessible materials allows students to study at their own pace beyond classroom hours, even in areas with unreliable internet connectivity. Furthermore, the platform enhances student engagement and promotes collaborative, project-based learning by offering real-time access to diagrams, video demonstrations, and software tutorials relevant to their courses. Ultimately, students are better prepared for examinations, practical projects, research activities, and industrial training experiences.

2. Faculty Empowerment and Innovation Culture

This project empowers the Faculty of Engineering to address its internal challenges using the very technical expertise it imparts to students. It demonstrates a bottom-up development approach, where faculties leverage engineering solutions rather than relying solely on external or government intervention to solve infrastructural and academic problems. The design,

development, deployment, and evaluation of the system provide practical engagement opportunities for both students and lecturers in areas such as systems integration, software customization, and digital infrastructure management. In doing so, the project fosters a culture of innovation, entrepreneurship, and problem-solving qualities essential for modern engineering professionals.

3. Scalability and Replication Across Institutions

A key strength of this project lies in its scalability. Built using low-cost, open-source tools and locally available resources, the system can easily be adapted for other faculties within the same university, including Science, Education, and Social Sciences. It can also be replicated in other universities, particularly those in rural or low-income settings, as well as in secondary and technical schools in underserved communities. This adaptability positions the project as a viable national model for integrating technology into education without the high infrastructure costs typically associated with digital transformation initiatives.

4. Bridging the Educational Digital Divide

One of the major inequalities in Nigerian education is the digital divide between students in urban, technology-enabled environments and those in underserved rural areas. Many students in public schools and low-income communities lack access to e-learning platforms, research databases, and learning management systems. This project provides a means to overcome these limitations through offline-capable digital libraries that can function without continuous internet connectivity. By reducing dependence on traditional infrastructure, the platform helps level the educational playing field, ensuring that students regardless of socioeconomic background have access to quality learning resources and modern academic tools.

5. Contribution to National Technological and Economic Development

A highly skilled engineering workforce is essential to Nigeria's progress in infrastructure, manufacturing, energy, transportation, and ICT. By improving the quality and accessibility of engineering education, this project contributes to the development of graduates who are technically competent, digitally literate, problem-oriented, and globally competitive. Such graduates are more likely to innovate, establish enterprises, enhance public infrastructure, and strengthen local industries, thereby advancing national development goals and supporting the growth of the Nigerian digital economy. Additionally, the project aligns with the objectives of the United Nations Sustainable Development Goal 4 (Quality Education), which promotes inclusive and equitable quality education and lifelong learning opportunities for all.

1.5 Scope of study

This study is focused on the Faculty of Engineering, University Of Benin and targets both students and lecturers across all departments within the Faculty of Engineering. It evaluates the effectiveness of the implemented system over a period of five to ten days following deployment. The study excludes real-time online systems due to existing constraints related to internet connectivity and power supply.

CHAPTER TWO

LITREATURE REVIEW

Technology integration in education refers to the effective use of digital tools and platforms such as computers, internet services, e-learning systems, and multimedia resources to enhance teaching, learning, and research. It allows for more interactive, flexible, and student-centered learning environments (UNESCO, 2022).

Globally, many higher education institutions have adopted technology to support academic delivery, enabling students to access virtual labs, digital libraries, online courses, and industry-relevant software tools. In particular, engineering education thrives on such integration, with simulations, CAD software, and virtual design platforms playing critical roles in building technical competence (Adebayo and Toyin, 2020).

Nigeria's educational system has made some efforts to embrace digital learning, but progress has been slow and uneven. According to Adeoye, Adanikin, and Adanikin (2020), most Nigerian universities lack the infrastructure and policy support necessary to implement large-scale e-learning. Many public institutions are characterized by outdated teaching practices, poorly equipped libraries, and limited internet connectivity.

In the engineering faculties, these problems are more glaring because of the field's dependence on technological resources. Tools such as AutoCAD, MATLAB, SolidWorks, and circuit simulation software are often unavailable or accessible only to a limited number of students. In some cases, engineering departments still rely solely on chalkboard lectures and photocopied handouts, despite the practical nature of their disciplines (Eze et al., 2021).

Studies from more developed contexts show that technology-based solutions improve academic engagement, knowledge retention, and skill acquisition in engineering programs. For instance, Oye et al. (2012) found that students exposed to e-learning platforms demonstrated stronger problem-solving abilities and were more confident using industry-relevant tools.

In Africa, several pilot projects have attempted to bridge the gap. One such initiative is the African Virtual University (AVU), which provides digital learning content for science and engineering programs across partner universities in over 20 African countries (AVU, 2020). Though useful, such solutions often require continuous internet access a challenge in most Nigerian settings.

2.1 Similar works include:

Kolibri is an open-source offline learning platform developed by Learning Equality in 2012 (Learning Equality, 2012). It was founded by Jamie Alexandre and Richard Culatta to expand access to quality education in low-connectivity environments (Alexandre & Culatta, 2012). The platform runs fully offline on low-powered devices such as Raspberry Pi and refurbished computers and allows synchronization through USB drives or occasional internet access. It supports multimedia educational resources including videos, eBooks, quizzes, and interactive simulations, and integrates open educational content from organizations such as Khan Academy and CK-12 Foundation (Khan Academy; CK-12 Foundation). Kolibri provides customizable content libraries, role-based teacher and learner accounts, progress tracking, analytics dashboards, peer-to-peer synchronization, and secure local network access. Although it has been piloted in parts of Nigeria for primary and secondary education, its application in higher engineering education remains limited.

The TELA (Technology-Enhanced Learning for All) Project was developed in 2019 by the Faculty of Education at the University of Ibadan (University of Ibadan, 2019). According to Olagunju and Adeleke (2020), the initiative demonstrated how faculty-led innovation could address teaching and learning challenges using localized digital tools. TELA adopted a blended learning approach, combining face-to-face classroom instruction with online modules, particularly benefiting pre-service and in-service teachers in distance and part-time programs. The project emphasized locally developed instructional materials aligned with Nigerian educational policies and pedagogy. It incorporated video lectures, PDFs, quizzes, assignments, discussion forums, and assessment tools, while also offering limited offline accessibility through preloaded storage devices. The system was implemented using open-source learning management systems such as Moodle (Moodle), enabling cost reduction, customization, and scalability.

Web-based course material management systems, commonly referred to as Learning Management Systems (LMS), have become central to digital education globally. One of the most widely adopted platforms is Moodle, developed by Martin Dougiamas in 2002 (Dougiamas & Taylor, 2003). Moodle is an open-source LMS grounded in social constructivist pedagogy, enabling educators to create structured courses, upload materials, administer assessments, and monitor student performance. Its flexibility, scalability, and cost-effectiveness have made it particularly attractive to institutions in developing countries. Research highlights Moodle's adaptability and strong community-driven development model as key success factors (Costa, Alvelos, & Teixeira, 2012).

Another prominent platform is Blackboard Learn, introduced in the late 1990s by Blackboard Inc. (Bradford, Porciello, Balkon, & Backus, 2007). Unlike Moodle, Blackboard is proprietary and

enterprise-focused, offering integrated tools for course content delivery, grading, virtual classrooms, and institutional analytics. Studies indicate that Blackboard has been widely adopted in North American and European universities due to its robust administrative features, though high licensing costs limit accessibility in low-income contexts (Coates, James, & Baldwin, 2005).

Canvas, developed by Instructure in 2011, represents a newer generation of cloud-based LMS platforms (Instructure, 2011). Canvas emphasizes user-friendly interface design, mobile compatibility, and seamless third-party integration. Research suggests that ease of use and intuitive navigation significantly influence LMS adoption and student satisfaction (Al-Busaidi & Al-Shihi, 2010). Canvas has gained popularity in higher education institutions seeking modern, scalable, and API-integrated systems.

In the open-source domain, Open edX, launched in 2012 by Harvard University and MIT (Agarwal, 2013), provides a large-scale platform for hosting Massive Open Online Courses (MOOCs). Open edX supports multimedia content, interactive problem sets, analytics dashboards, and discussion forums. Its architecture allows institutions to customize learning experiences while maintaining scalability for thousands of users. Studies on MOOCs highlight their potential for widening access to higher education, though challenges remain in learner retention and engagement (Jordan, 2014).

Additionally, ATutor, developed in 2002 at the University of Toronto (Vassiliou & McAndrew, 2004), focuses on accessibility and inclusive design, complying with web accessibility standards. ATutor has been particularly relevant for institutions prioritizing learners with disabilities. Similarly, Claroline, initiated at the Université catholique de Louvain in 2001 (Claroline

Consortium, 2001), supports collaborative course creation, document sharing, and communication tools within academic environments.

Across these platforms, research consistently identifies critical success factors such as institutional support, digital literacy, infrastructure readiness, and curriculum alignment (Bhuasiri et al., 2012; Andersson & Grönlund, 2009). While these systems demonstrate the global evolution of web-based course material management, their effectiveness in developing-country contexts depends heavily on sustainability planning, localized customization, and offline compatibility.

Engineering ELA is a web-based platform designed to provide structured digital solutions to over 30 Engineering Laboratory Assistant (ELA) practicals. It focuses specifically on engineering students and educators, enhancing laboratory instruction through organized digital experiment guides and accessible learning materials. Unlike Kolibri and TELA, which serve broader educational contexts, Engineering ELA is tailored to engineering laboratory education, aiming to improve practical comprehension, preparation, and overall laboratory performance.

2.2 Challenges identified in prior efforts

Although numerous technology-integration initiatives have been introduced across Nigeria and sub-Saharan Africa particularly in response to the educational disruptions caused by the COVID-19 pandemic and widening digital inequality (UNESCO, 2020; World Bank, 2021) significant research and implementation gaps remain. Many of these interventions have either underperformed or failed to achieve sustainable impact, especially within higher education and engineering faculties (Adarkwah, 2021; Dhawan, 2020). Existing studies largely focus on basic education or generalized e-learning systems, with limited emphasis on discipline-specific,

offline-capable, and infrastructure-sensitive solutions tailored to engineering education (Bhuasiri et al., 2012; Andersson & Grönlund, 2009). Furthermore, gaps persist in areas such as long-term sustainability, localized content development, technical capacity building, and evaluation of short-term post-deployment effectiveness in developing-country contexts (Unwin, 2009; OECD, 2020). These shortcomings highlight the need for a more context-aware, faculty-driven, and scalable approach to digital academic resource management.

Numerous technology-driven education initiatives across Nigeria and sub-Saharan Africa have struggled due to weak sustainability and maintenance planning. Several studies have identified the absence of long-term funding models, inadequate technical support personnel, and lack of structured maintenance strategies as major contributors to project failure (Unwin, 2009; Andersson & Grönlund, 2009). Many donor-funded ICT projects declined after initial implementation because there were no recurring budget allocations for system upgrades, hardware replacement, software updates, or content refresh (World Bank, 2021). Without clearly defined sustainability frameworks, digital education systems often become obsolete within a few years of deployment.

Poor infrastructure, particularly unstable electricity supply and unreliable internet connectivity, has also significantly undermined educational technology initiatives in Nigeria. Reports by UNESCO (2020) and the World Bank (2021) highlight that frequent power outages and limited broadband penetration restrict the consistent use of online learning platforms. Dhawan (2020) further observed that the rapid transition to fully online systems during COVID-19 exposed severe infrastructural weaknesses in developing countries. Many platforms were designed with constant internet connectivity in mind, making them impractical in institutions lacking robust IT

infrastructure. These challenges emphasize the importance of offline-capable and energy-efficient solutions in low-resource settings.

Limited stakeholder involvement has likewise affected the success of many projects. Research shows that top-down implementation models where lecturers, students, and institutional IT staff are excluded from planning often result in low adoption rates and resistance (Bhuasiri et al., 2012; Adarkwah, 2021). When digital systems are not aligned with curriculum needs or when users feel imposed upon, engagement declines. Successful implementations, therefore, require participatory design approaches that involve end-users in development, testing, and deployment stages.

Digital literacy gaps further constrain technology adoption in higher education. Despite university-level enrollment, many students and lecturers lack the practical skills required to effectively navigate learning management systems and digital libraries (Andersson & Grönlund, 2009; OECD, 2020). Studies indicate that without structured training and continuous user support, even well-designed platforms experience declining usage after initial introduction (Adarkwah, 2021). Complex interfaces and weak integration into classroom pedagogy often lead to abandonment of digital tools.

Content misalignment and copyright constraints represent another recurring challenge. Several interventions deployed generic or foreign educational content that did not align with local curricula or national academic standards (Unwin, 2009; UNESCO, 2020). This disconnect limits relevance, particularly in technical disciplines like engineering, where localized standards, practical guides, and context-specific case studies are essential. Additionally, copyright

restrictions frequently limit the digitization and sharing of essential textbooks and journals, reducing the overall effectiveness of digital repositories (World Bank, 2021).

The adoption of one-size-fits-all solutions has also reduced impact, especially in STEM fields. Engineering education requires access to simulations, design tools, laboratory guides, and technical documentation that differ substantially from the needs of other faculties. Bhuasiri et al. (2012) emphasize that discipline-specific customization is a critical success factor in e-learning implementation. General-purpose content platforms without engineering-focused features often fail to meet the practical and technical demands of such programs.

Finally, inadequate monitoring and evaluation mechanisms have limited continuous improvement in many ICT education initiatives. Projects frequently lack performance metrics, user feedback systems, and data-driven refinement processes (OECD, 2020; World Bank, 2021). Without systematic assessment of usage patterns and academic impact, platforms stagnate and fail to evolve in response to user needs.

Collectively, these documented challenges demonstrate that many past interventions were overly ambitious, underfunded, insufficiently localized, or disconnected from institutional realities (UNESCO, 2020; Unwin, 2009). A sustainable and effective solution must therefore be low-cost, offline-compatible, and scalable; incorporate structured training and capacity-building; provide engineering-relevant and curriculum-aligned content; adopt a participatory design approach involving faculty and students; and include a clearly defined long-term sustainability and evaluation framework.

2.3 Research gaps identified

Despite the growing global emphasis on educational technology, there remains a significant gap in both literature and practical implementation when it comes to addressing the unique needs of engineering education in Nigeria. Specifically:

I. Very few studies or projects focus exclusively on the challenges faced by engineering faculties in Nigerian universities, particularly in terms of digital access to course materials and technical resources.

II. Most existing solutions are internet-dependent, which makes them unsuitable for environments with unstable connectivity. There is a lack of offline, self-hosted solutions that can serve the needs of STEM students in low-resource settings.

III. Moreover, previous interventions are rarely designed and implemented by engineering students or faculties themselves. This limits practical engagement, sustainability, and local ownership.

This study directly addresses these gaps by proposing a localized, faculty-specific digital library system one that is affordable, offline-accessible, and built through an engineering-led, in-house approach. By empowering students within the Faculty of Engineering to create a system tailored to their curriculum and resource challenges, the project offers a scalable and replicable model that other faculties and institutions across Nigeria can adopt.

CHAPTER THREE

METHODOLOGY

3.0Introduction

After analyzing previous initiatives aimed at improving access to educational resources such as internal repositories and offline databases we observed that while these solutions were helpful, they often lacked user-centered features such as intuitive navigation, real-time updates, and download flexibility. These limitations highlighted the need for a more dynamic and accessible system. Based on feedback from our peers and a review of digital learning trends, my team and I concluded that engineering students would benefit significantly from a feature-rich web application that not only allows them to view and download course materials but also ensures easy access, categorization by course codes.

To ensure our design was informed by actual user needs and representative data, we began by calculating the sample size required for our survey. This step was critical in making evidence-based decisions throughout the development process. We first estimated the total student population within the Faculty of Engineering at the University of Benin and then applied the standard statistical formula for sample size determination. This allowed us to define a reliable number of respondents needed to capture diverse user preferences and challenges related to accessing educational content.

The formular we used for calculating sample size;

$n = N/1 + N(e)^2$

Where:

I. n = sample size

II. N = population size

III. e = sample error (5%)

a) $n = 5000/1 + 5000(0.05)^2$

b) n = 370.37~370 sample size

Eq 2.1 sample size formular

Following the determination of our required sample size, we proceeded to gather the necessary contact information to facilitate effective data collection across the Faculty of Engineering. To achieve this, we reached out to all class representatives and faculty executive members EXCOS, who serve as key communication links within the student body. We shared the objectives of our project with them, and upon understanding its significance, they expressed their support and assisted in compiling the contact details of students across various departments and levels.

The collected data including names, departments, academic levels, and positions held was carefully documented and systematically organized to streamline communication and ensure comprehensive outreach. With this groundwork completed, we designed a closed-ended, multiple-response questionnaire in a checkbox format. This survey instrument was structured to capture student opinions, preferences, and challenges regarding access to educational materials and the integration of technology within the faculty. The closed-ended format ensured consistency in responses, making the data easier to analyze and interpret.

Once the questionnaire was finalized and approved by our project supervisor, we proceeded to design a visually appealing digital flyer to promote participation and attract attention from students across the Faculty of Engineering. This flyer contained a brief overview of the project, a call to action, and a link or QR code directing respondents to the survey form. The goal was to encourage maximum participation through a familiar and accessible channel.

Within the span of one week, we received over 600 valid responses significantly exceeding our required sample size. This strong level of engagement not only validated the relevance of our research focus but also provided a diverse and rich dataset for analysis. Upon completing data analysis, we identified key trends, needs, and feature requests expressed by the students. These insights informed the next stage of development, where we began crafting a detailed user story. The user story captured the expectations, behaviors, and challenges of our target users, and served as a foundational guide for the design and development of the web-based course material management system. See appendix for questionnaire.

3.1Project Planning and Approval Phase

After identifying the problem discussed in Chapter One, our team proceeded to the Project Planning and Approval Phase, which served as the foundation for the entire project. At this stage, we developed a comprehensive project plan that clearly outlined how the team intended to approach and execute the development of the UnibenEngVault platform. The project plan covered all key aspects necessary for successful implementation, including the project scope, objectives, timeline, team roles and responsibilities, cost and resource analysis, evaluation criteria, risk management strategies, expected deliverables, and communication framework. Each component was carefully discussed and documented to ensure that every team member understood their tasks and how they contributed to the overall project goals.

The plan was formally documented in a shared Google Docs file titled “Final Year Project Plan”, which served as the central reference document for the entire team. This approach allowed for real-time collaboration, version tracking, and easy access, ensuring that updates or revisions were immediately visible to all members.

To support financial accountability and effective monitoring, the team created a detailed budget and expense tracking sheet, which recorded all costs associated with materials, software tools, and other project-related expenses. In addition, a meeting attendance and activity log was maintained to document participation, track decisions made during meetings, and follow up on assigned tasks.

To enhance time management and ensure progress transparency, the team developed a Gantt chart using Google Sheets. The Gantt chart displayed the project timeline, milestones, and task dependencies, making it easier to visualize the project’s progression and identify overlapping activities or potential delays. Regular updates were made to reflect completed tasks and upcoming deadlines.

All planning documents including the project plan, budget sheet, attendance log, and Gantt chart were neatly organized within a dedicated Google Drive folder shared among team members. This central repository streamlined collaboration, improved accessibility, and reduced the risk of data loss or miscommunication. The use of cloud-based tools also allowed for continuous monitoring, ensuring that everyone remained aligned and up to date regardless of their location.

Overall, the planning and approval phase provided a strong organizational foundation for the project. It ensured that every activity was well-documented, responsibilities were clearly defined, and the project could proceed in a structured, transparent, and goal-oriented manner.

3.2Understanding the Problem Phase

3.2.1 Problem Solving Worksheet

Since we already had a general idea of the problem we were trying to solve, this phase focused on diving deeper into the details. We wanted to identify the exact group of students affected by the lack of a central platform for accessing study materials. To achieve this, we used our team's Problem-Solving Worksheet to guide our analysis. The worksheet helped us answer key questions such as: What is the problem? Who does the problem impact, directly or indirectly? When did this problem begin? When does it occur? Where is this problem occurring? Why is this a problem? How would reality be different if this problem were solved? Are there other benefits that would come from the problem being solved? Approximately how many students are directly impacted by this problem? What else would you like to know or understand better about the problem?

A screenshot of our problem-solving worksheet is shown in figure 3.1. Each team member had an editable copy to fill and submit on our team's platform. The idea of giving everyone an individual copy was to help us think deeply and independently about the problem. After all submissions, we carried out a peer review, where each member reviewed another's responses to gain more insights into different perspectives. This process helped us broaden our understanding and learn from one another's experiences regarding the problem.

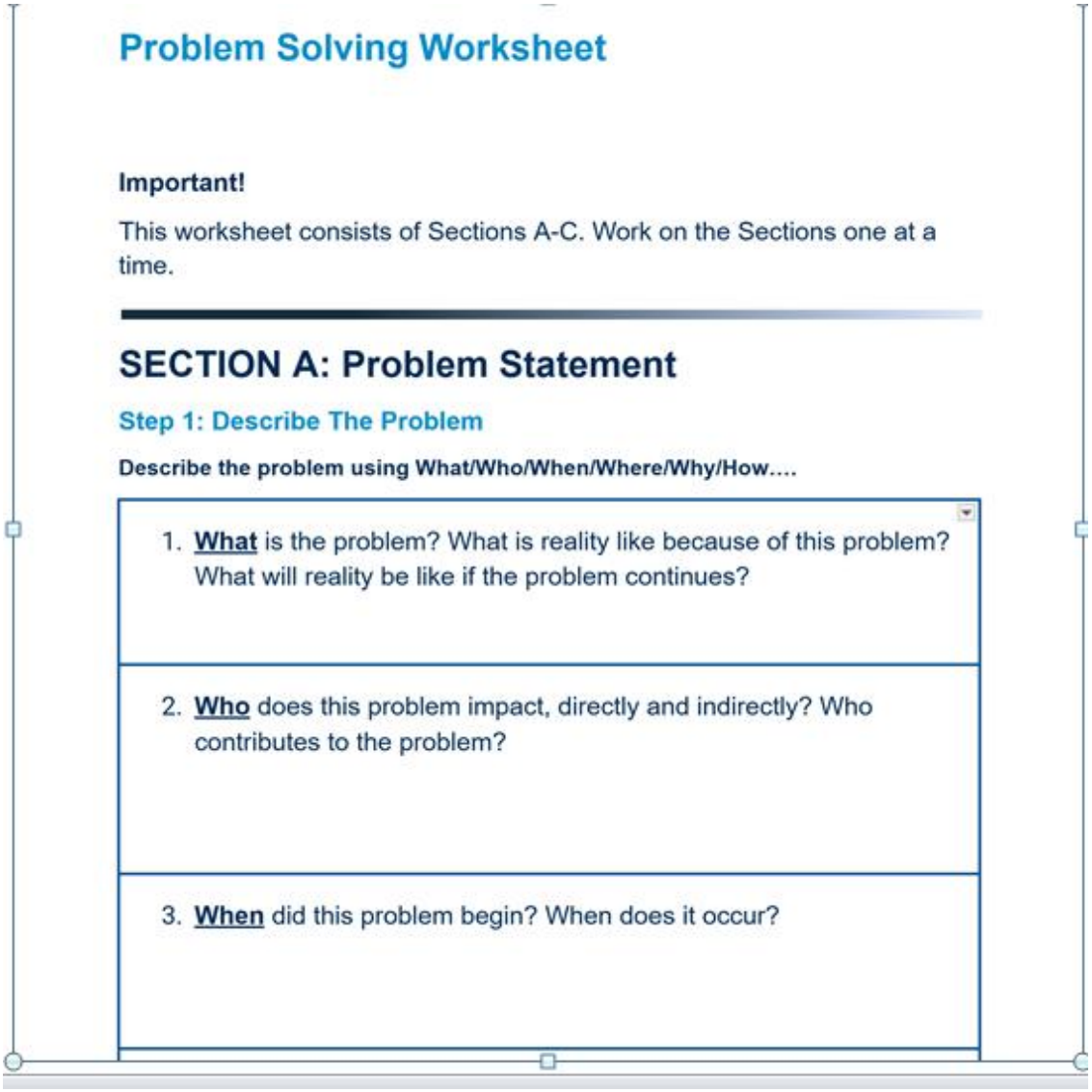


Figure 3.1: Screenshot of the team's problem-solving worksheet

3.2.2 Research Worksheet

After completing the Problem-Solving Worksheet, we used the Research Worksheet to focus on what we still wanted to discover about the problem. Each team member came up with three research questions, and from all of these, we carefully compiled a final set of questions for our questionnaire. The questionnaire helped guide our research and gather information directly from the students who are affected by the problem. The questions acted like hypotheses assumptions we had about the problem and hearing directly from the students helped us understand it from their perspective. In the Research Worksheet, we defined:

I. The objectives of our research

II. Three research questions and associated hypothesis

III. The type of data needed (quantitative or qualitative)

IV. Where and how to collect the information (e.g., directly from students, online sources, or asking lecturers)

A screenshot of our research worksheet is shown in figure 3.2. This document helped the team clearly plan the data collection process and ensured that our research would answer the most important questions about the problem.

Research Worksheet

You can plan out your research for each of your research questions using the below template.

PART A

Research Step	Key Questions	Your Responses
Define your objectives, 3 research questions & associated hypotheses.	What are you trying to accomplish with this research? What do you wish to find out that will help in solving the problem?	
Determine what kind of data you need.	Think about whether you need numbers (quantitative) or opinions and experiences (qualitative). Will you use surveys, interviews, or forms? In what format do you want to present your findings - maybe a simple report, slide?	
Think about where to get the information.	Do you already know people you want to talk to (like students or lecturers)? Are there existing sources you can check out?	

Figure 3.2 : Screenshot of the team's Research Worksheet

3.2.3Questionnaire Design and Data Collection

With the research questions set, we created a questionnaire consisting of twelve closed-ended questions. The questionnaire was designed and managed using Google Forms. To ensure we efficiently reached the entire Faculty of Engineering, we collected and documented the contacts of all course representatives, assistant course reps, class reps, departmental executives, and faculty executives NUESA (Nigerian Universities Engineering Student's Association).

We shared the questionnaire with them and requested their help in distributing it to their class group chats. Within one day, we gathered 660 responses from engineering students, which exceeded the number of students we had initially targeted. The responses covered students across all departments and levels.

After data collection, we combined all response sheets into a single spreadsheet and filtered out non-engineering student responses, leaving about 640+ responses for analysis. The filtration ensured that our focus remained solely on engineering students, who are the intended users of the platform.

We then conducted data analysis to identify:

1. The number of students interested in having such an application
2. The features students wanted most on the platform
3. The types of study materials students wanted to see on the platform

Screenshots of the data analysis report are shown below. These visuals highlight the insights that guided our platform design and feature prioritization.

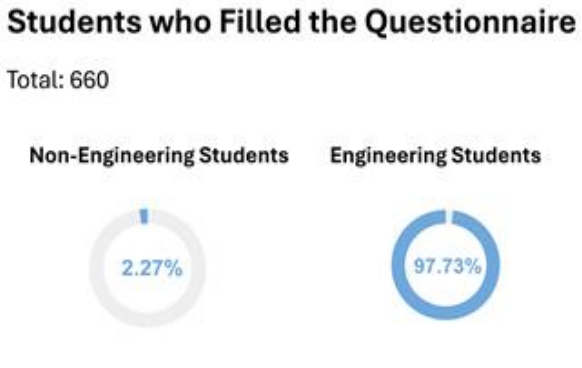


Figure 3.3: Students who filled the questionnaire (engineering vs non-engineering)

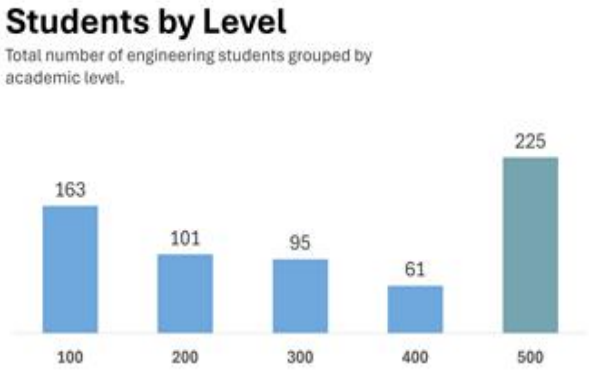


Figure 3.4: Students by level (100-500)

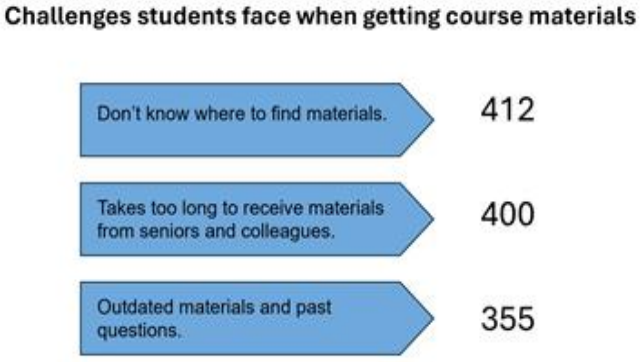


Figure 3.5: Challenges students face getting course materials.

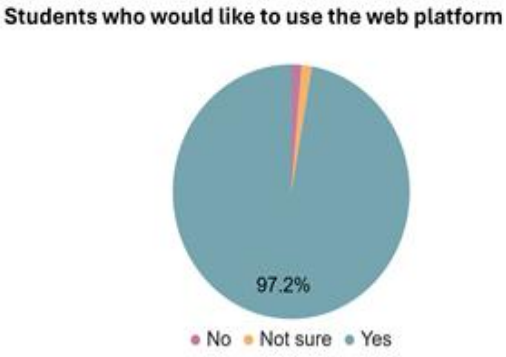


Figure 3.6: Students who want to use the platform

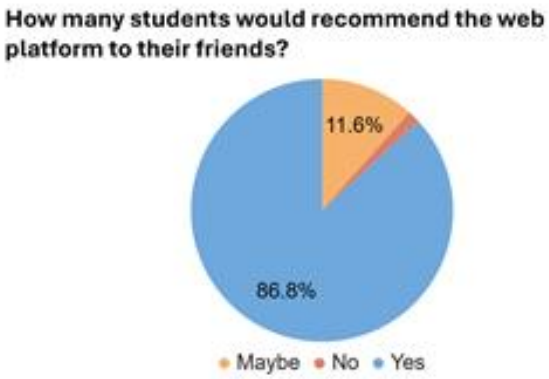


Figure 3.7: Students who would recommend the platform

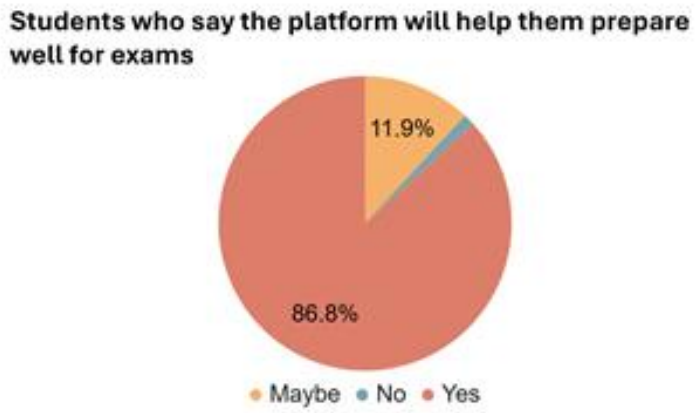


Figure 3.8: Platform helping students prepare for exams

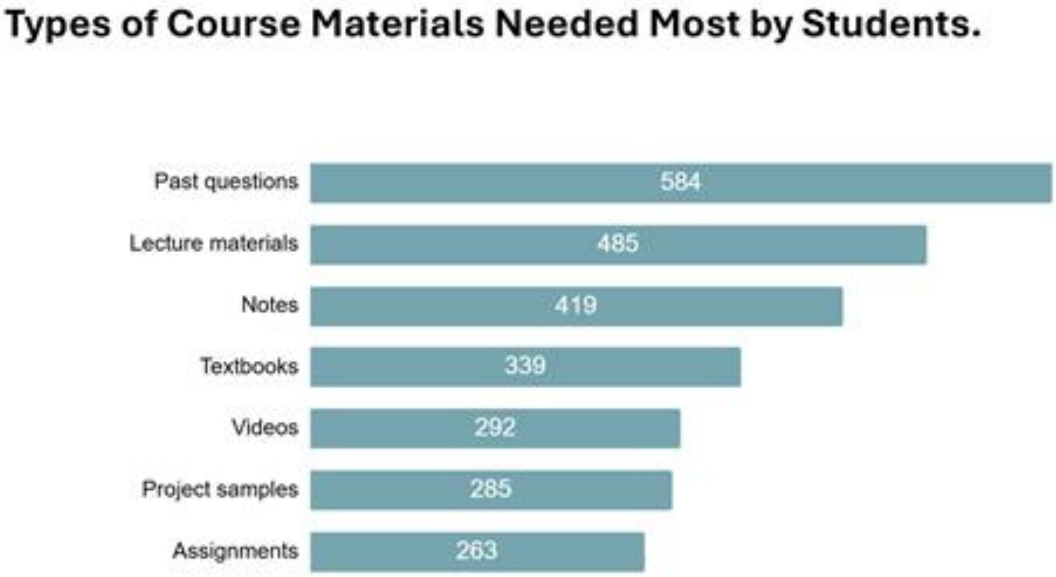


Figure 3.9: Most needed course material

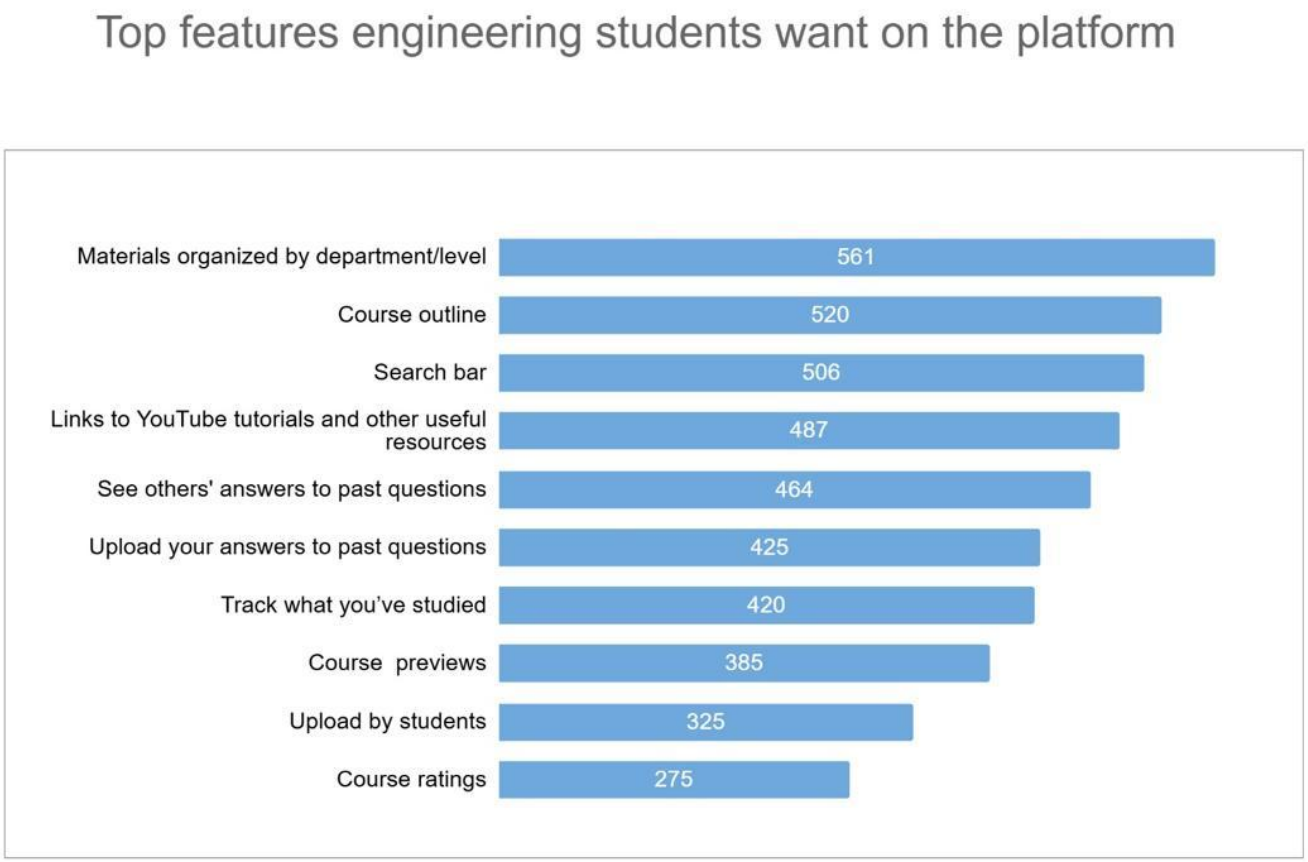


Figure 3.10: Top features students want

3.2.4User Story Worksheet

From the questionnaire responses, we were able to confirm that the problem we are trying to solve is valid, as almost all engineering students expressed interest and excitement about the platform. At this stage, we had a clear understanding of both the problem and how to solve it in a way that meets students' needs.

Next, we filled out our User Story Worksheet. User stories are a simple way to describe what a user (student) wants to do on the platform and why. For example, if a student mentioned that they struggle to find past questions, we turned that into a user story to guide design.

User stories helped us think from the students' perspective, not just as software developers or planners. Each user story includes acceptance criteria, which lets us know whether it has been implemented correctly.

Example of a user story:

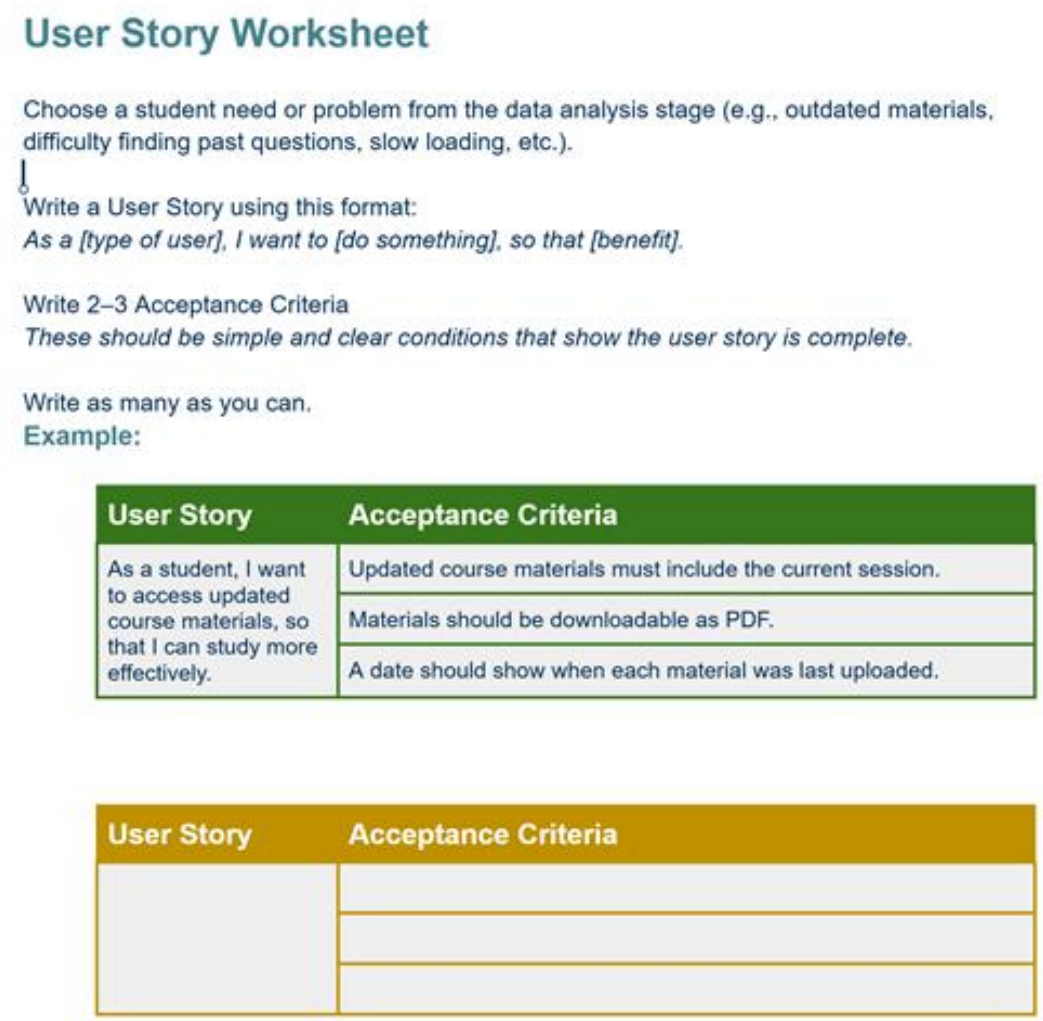
User Story:

As a student, I want to be able to search for past questions easily so I can prepare better for my exams.

Acceptance Criteria:

1. The search bar appears on the homepage.
2. The student can type a course name into the search bar.
3. The correct past questions show up when they search.

A screenshot of our User Story Worksheet is shown below:



User Story Worksheet

Choose a student need or problem from the data analysis stage (e.g., outdated materials, difficulty finding past questions, slow loading, etc.).

Write a User Story using this format:
As a [type of user], I want to [do something], so that [benefit].

Write 2–3 Acceptance Criteria
These should be simple and clear conditions that show the user story is complete.

Write as many as you can.
Example:

User Story	Acceptance Criteria
As a student, I want to access updated course materials, so that I can study more effectively.	Updated course materials must include the current session.
	Materials should be downloadable as PDF.
	A date should show when each material was last uploaded.

User Story	Acceptance Criteria

Figure 3.11: Screenshot of the team's User Story Worksheet

By the end of the Understanding the Problem phase, we had a clear picture of the problem we were solving and the students' needs. The Problem-Solving Worksheet, Research Worksheet, Questionnaires, and User Story Worksheet helped us identify key features and requirements for the platform. Regular team meetings ensured everyone stayed aligned. This understanding formed a solid foundation for the Design Phase, where we translated insights into wireframes, UML diagrams, and the system architecture.

3.3Design Phase

In this phase, we translated our user stories into user roadmaps and wireframes. This stage also included choosing colours, fonts, and screen layouts for the user interface (UI). We created UML (Unified Modelling Language) diagrams to visualize the web application system and an ERD (Entity-Relationship Diagram) to guide our database design. To ensure everyone was aligned, we also held several mini-classes to explain key software engineering concepts to the team members.

3.3.1Wireframe Challenge and Prototyping

Using the user stories, we created user roadmaps that outlined the steps a student would take while using the platform. From these roadmaps, we were able to identify the actual web pages needed to support each part of the student's journey. This approach ensured that every page had a clear purpose and addressed real students' needs. The web pages included: Home page, Registration page, Login page, My courses page, Files page, Profile page, Admin's dashboard page, Manage users page, Manage departments and levels page, Manage courses page, Manage file uploads page, Manage tutorial links page.

After creating the user roadmaps, we moved on to the wireframe challenge. A wireframe is like a rough sketch of a web application. It shows the basic layout of each page where buttons, menus, images, and text will go. Wireframes focus on functionality and navigation, not colors or design.

During the wireframe challenge, each team member took a page to design and then presented it to the rest of the team. This was an exciting experience because we learned how web pages work individually and how they come together as a complete web application, with navigation guiding students from one page to another with just a click.

3.3.2 Design Choices: Colors, Fonts, and Screen Sizes

After completing the wireframe challenge, we made some preliminary design decisions for the platform, focusing on colours, fonts, and screen sizes to guide the eventual UI/UX development.

We chose the following colours:

1. Primary (base) colour: Purple
2. Secondary colour: Blue
3. Grey colors: White and Black

For fonts, we selected Poppins for a clean and modern look. We also considered different screen sizes. For students, we planned for both phone and laptop screens, since most students primarily use their phones for studying, ensuring accessibility and inclusivity. For admin management, we focused on laptop screens, as administrative tasks typically require larger displays for efficiency.

3.3.3 UI/UX - User Interface and User Experience

UI (User Interface) is about how the platform looks and how all the elements are arranged on the screen. It shows the layout, sizing, spacing, colors, and placement of buttons, menus, forms, and other components. In other words, it is the visual plan that guides frontend developers in building the pages. Think of it like an architectural drawing for a building without it, the developers would place elements randomly, and the final design could look messy, confusing, or hard to use.

A good UI ensures that:

1. Students can easily find what they need without unnecessary searching.
2. Buttons and links are visible and accessible, reducing frustration.
3. Colors, fonts, and spacing create a pleasant and professional look.

UX (User Experience) focuses on how the user feels when interacting with the platform. It ensures that the application is:

1. Intuitive: Students immediately understand how to navigate the app.
2. Efficient: Students can accomplish tasks, like finding past questions or downloading materials, quickly.
3. Consistent: Similar actions behave the same way throughout the platform.
4. Accessible: Students with different devices, screen sizes, and even visual abilities can use the platform comfortably.

In UnibenEngVault, UI and UX work together to create a balanced experience. The UI provides a clear structure and visually appealing layout, while UX ensures that students don't feel lost, frustrated, or overwhelmed. For example:

1. The homepage clearly shows courses and notifications in organized sections.
2. Buttons like "Download" or "View File" are placed logically and are easy to click.
3. Color choices, fonts, and spacing are selected to be pleasant on both laptop and phone screens, making sure most students can use the app easily since many rely on mobile devices.

By carefully planning both UI and UX, we ensured that the platform is not too complex to confuse users and not too simple to look boring. It creates a smooth and enjoyable learning experience, encouraging students to use the platform regularly.

3.3.4 UI/UX Technology Used: Figma

The design of the UnibenEngVault interface was created using Figma, a powerful, cloud-based design and prototyping tool widely recognized for its collaborative capabilities and efficiency in UI/UX development. Figma was chosen because it allows designers, developers, and other team members to work together in real-time, ensuring that feedback and updates happen seamlessly without the need for multiple file versions. Major reasons for using Figma include:

1. **Interface Design and Layout Creation**

   The lead designer used Figma to create detailed frames and artboards that represented each screen of the platform such as the homepage, course pages, file sections, and login interfaces. Grids and layout guides were applied to maintain visual consistency, proper

spacing, and alignment across all pages. These layout frameworks ensured that the platform remained well-structured, scalable, and adaptive to different screen sizes (desktop, tablet, and mobile).

2. **Typography, Colors, and Components**

Through research and design experimentation, appropriate font styles, color palettes, and component sizes were selected to reflect a professional yet friendly academic environment. The color scheme was carefully balanced to enhance readability and aesthetic appeal, while ensuring accessibility for users with visual impairments.

Figma's component and style system allowed the designer to create reusable elements such as buttons, icons, and navigation bars. This not only improved design consistency but also made future updates faster and easier.

3. **Prototyping and User Flow Testing**

Figma's interactive prototyping feature was used to simulate the user experience before actual development began. Links were created between pages to represent realistic navigation paths, allowing the team to test and refine the user flow. This stage helped identify potential usability challenges early, ensuring that students could move smoothly through the platform without confusion or dead ends.

4. **Collaboration and Team Input**

One of Figma's biggest advantages is its real-time collaboration feature. Team members could view the design process live, leave comments, and suggest modifications instantly. This eliminated communication delays and made it easier for the developers to understand how each UI element should behave once implemented.

Additionally, the design process involved regular design reviews, where team feedback was incorporated to refine layouts, improve contrast, and optimize visual hierarchy.

5. **Design Inspiration and Research**

The designer leveraged Pinterest, Dribbble, and Behance to gather creative ideas and current design trends in educational technology platforms. These inspirations helped in choosing modern yet functional design styles that appeal to students while maintaining simplicity and clarity.

Usability principles such as color psychology, visual hierarchy, and minimalist design were applied to ensure the platform remained user-centered and engaging.

6. **Developer Handoff and Implementation Support**

Once the designs were finalized, Figma's developer handoff tools (such as inspect mode and CSS code snippets) provided the frontend team with accurate measurements, color codes, and spacing values. This made it easier to translate the visual designs into working web pages without guesswork or inconsistencies.

figure3.12 and 3.13 show screenshots of the designs created using figma include:

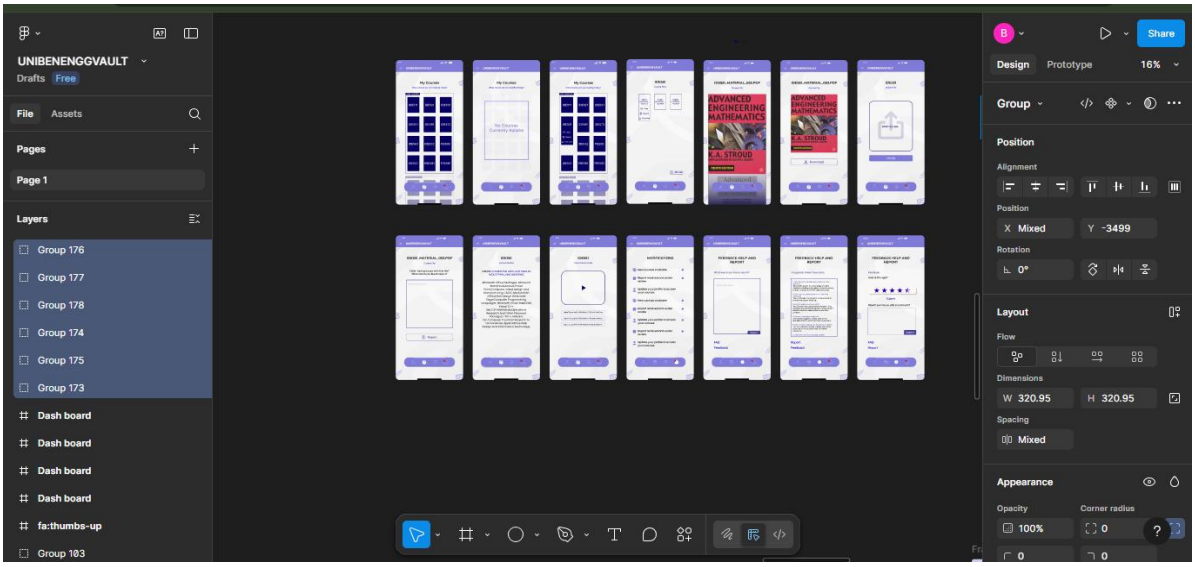


Figure 3.12: Screenshot of mobile app interface design

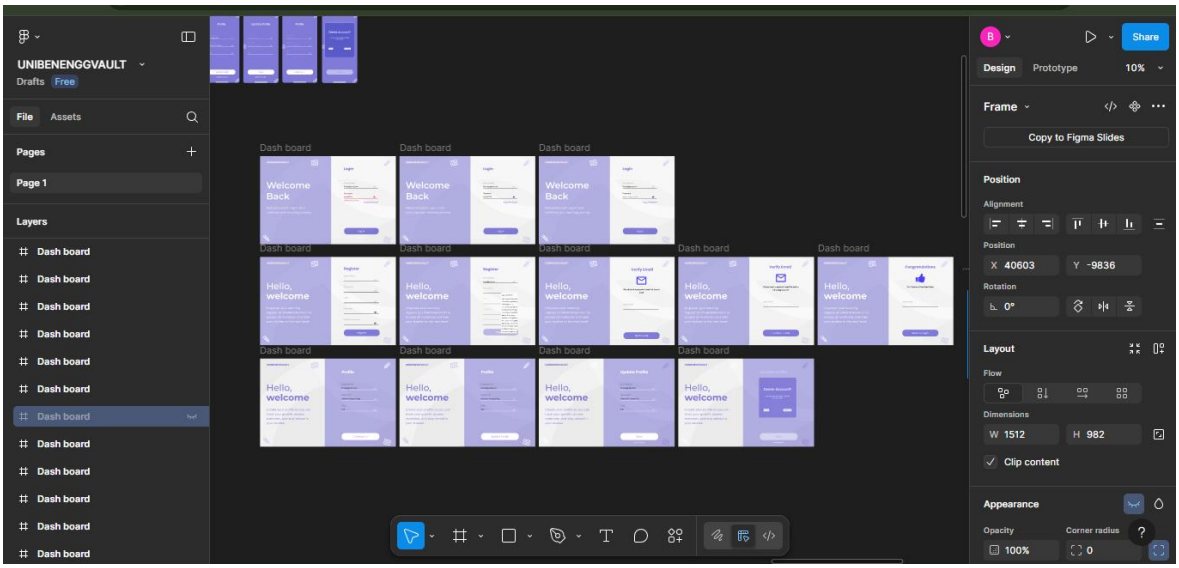


Figure 3.13: Screenshot of desktop user interface design

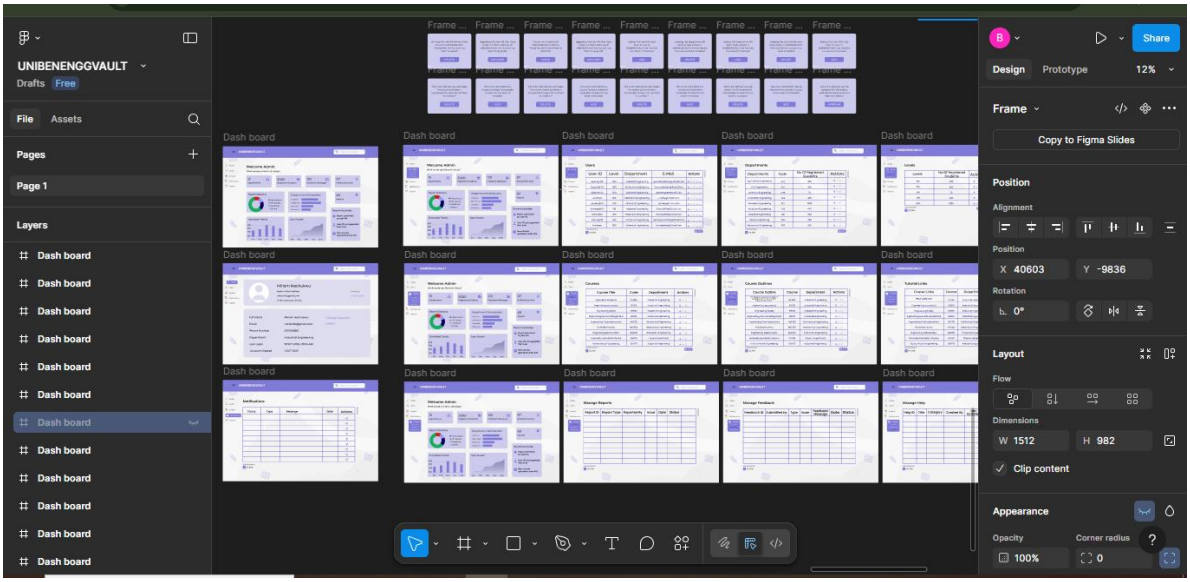


Figure 3.14 Screenshot of admin user interface design

3.3.5 UML Diagram and ERD

UML (Unified Modeling Language) diagrams help visualize the structure and behavior of the system. They show how users interact with the system, the different components, and how the components relate to each other as shown in figure 3.15 and figure 3.16.

ERD (Entity-Relationship Diagram) shows the structure of the database and how information like students, courses, and files are connected as shown in figure 3.17 and figure 3.19.

Figure3.15: Student use case diagram

Figure3.16: Admin use case diagram

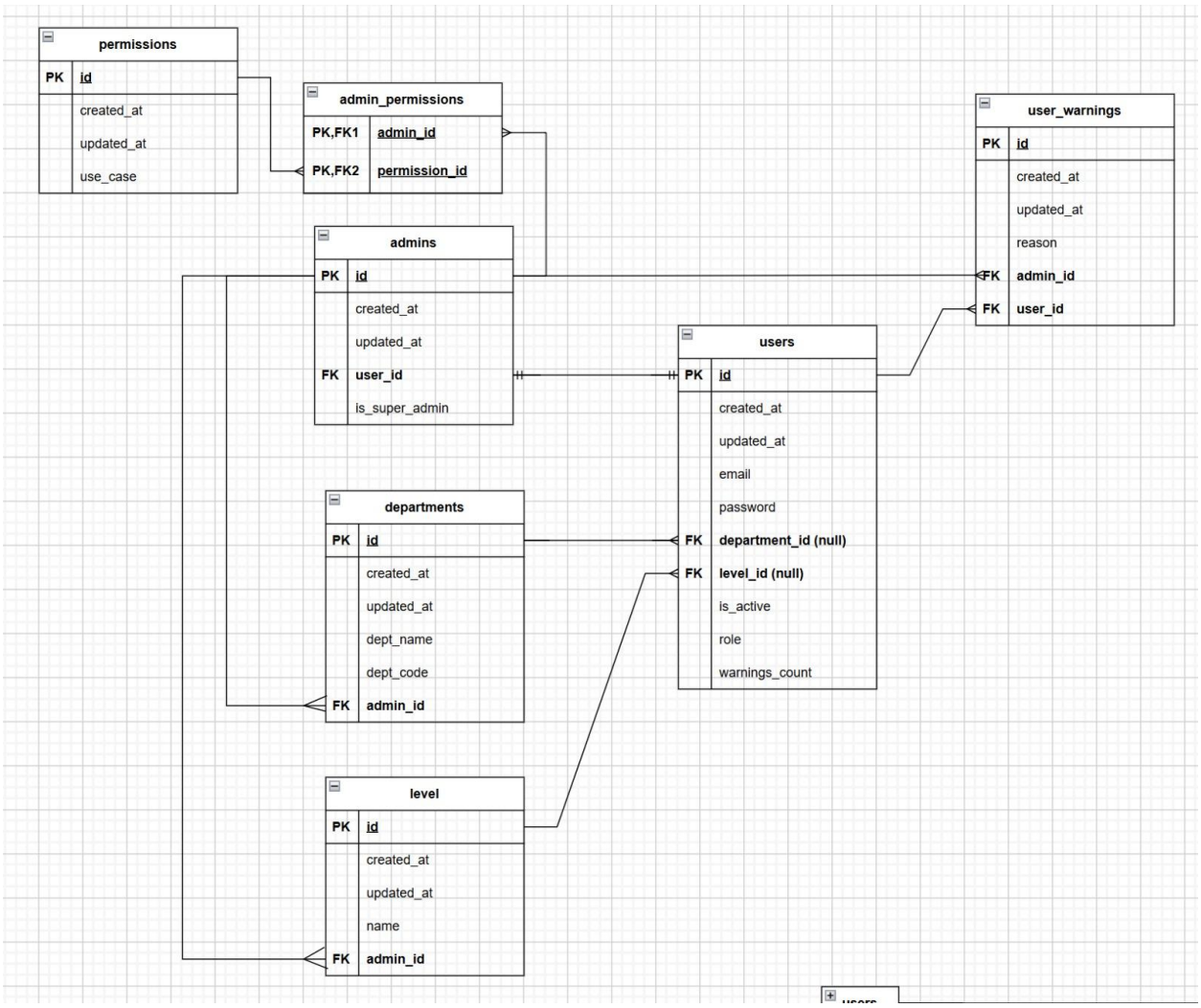


Figure3.17: ERD (Entity Relationshp Diagram) showing admins. Permissions and users tables

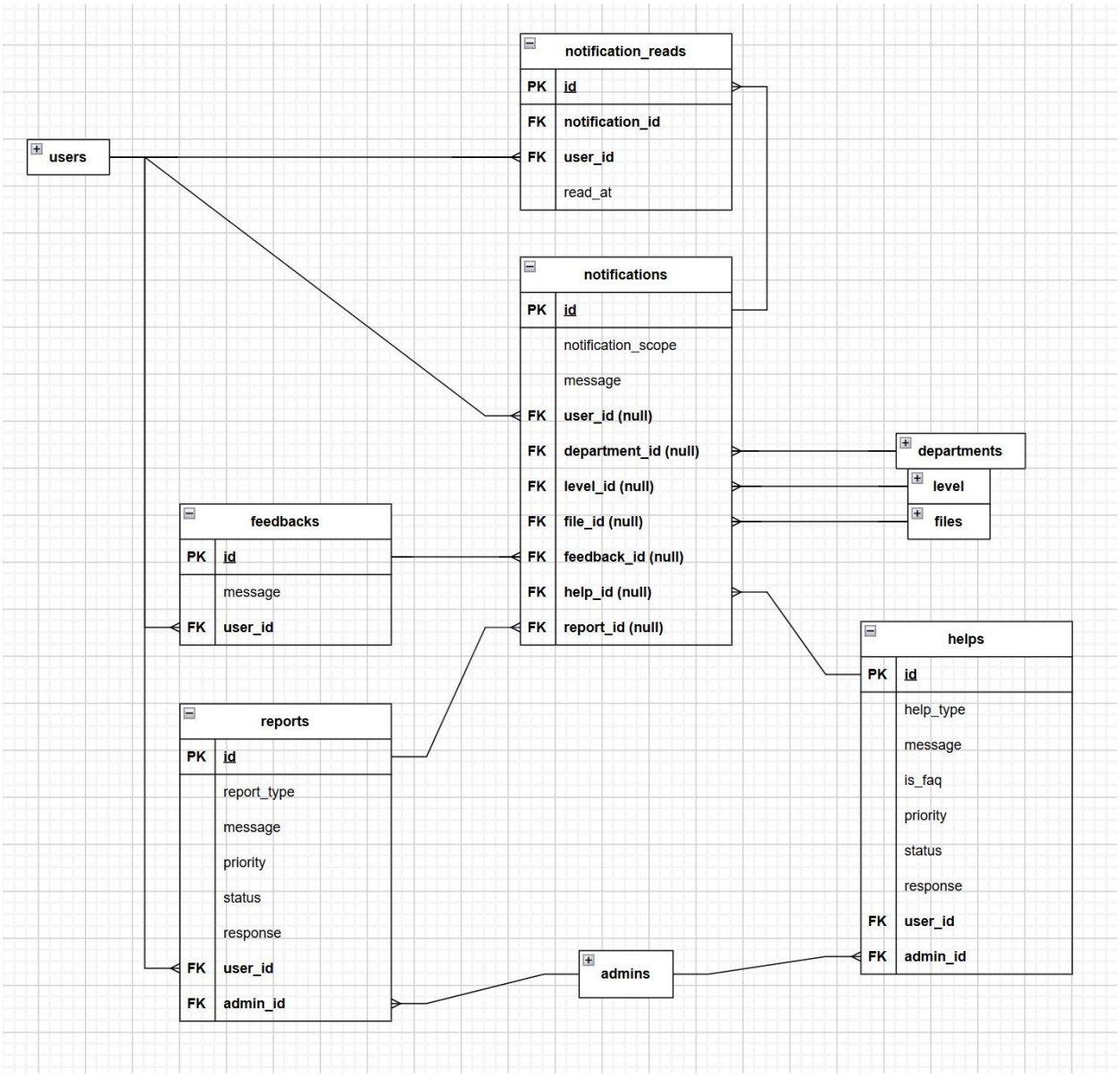


Figure 3.18 ERD Showing courses, files department and levels tables.

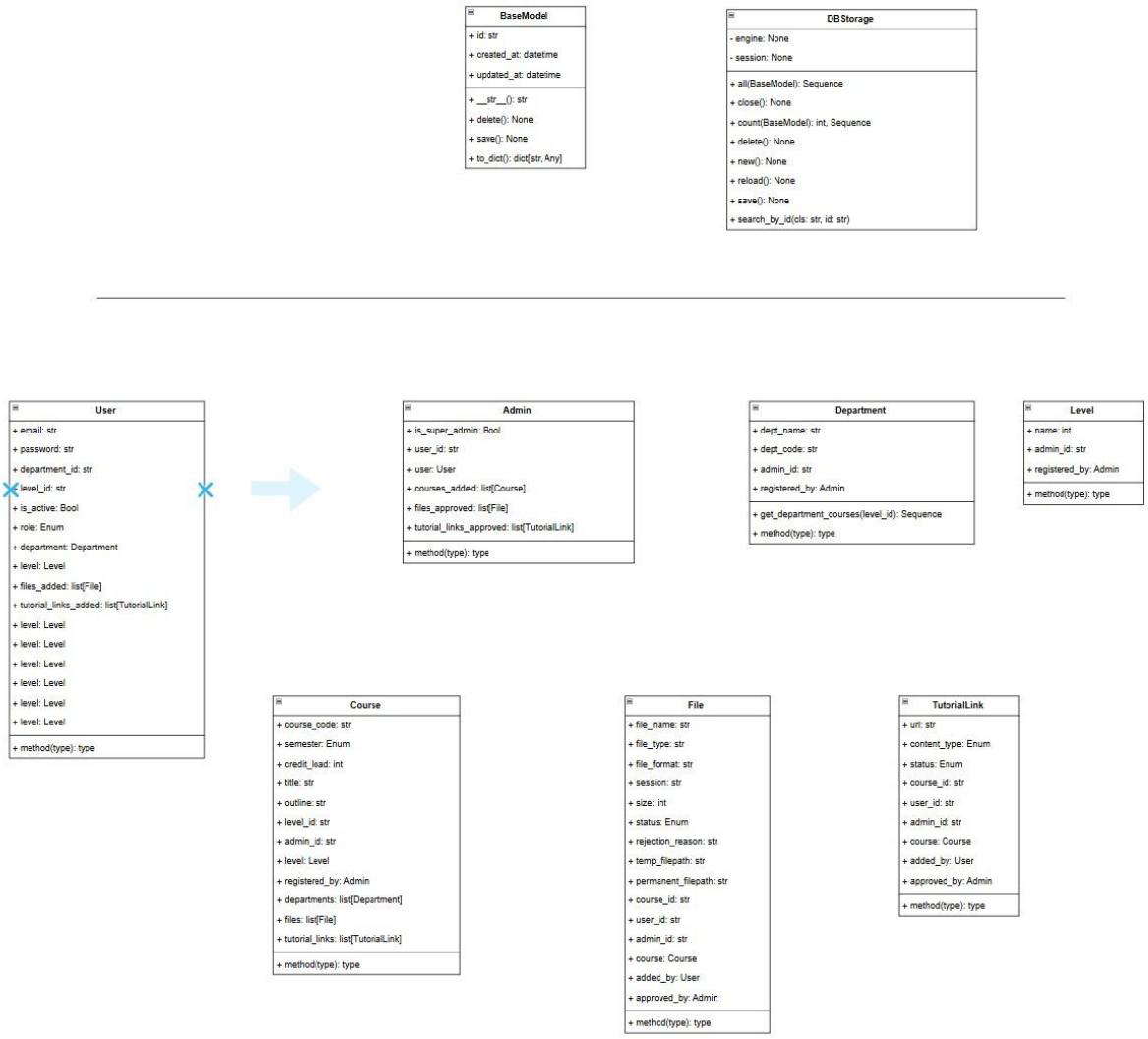


Figure 3.19 Class diagrams

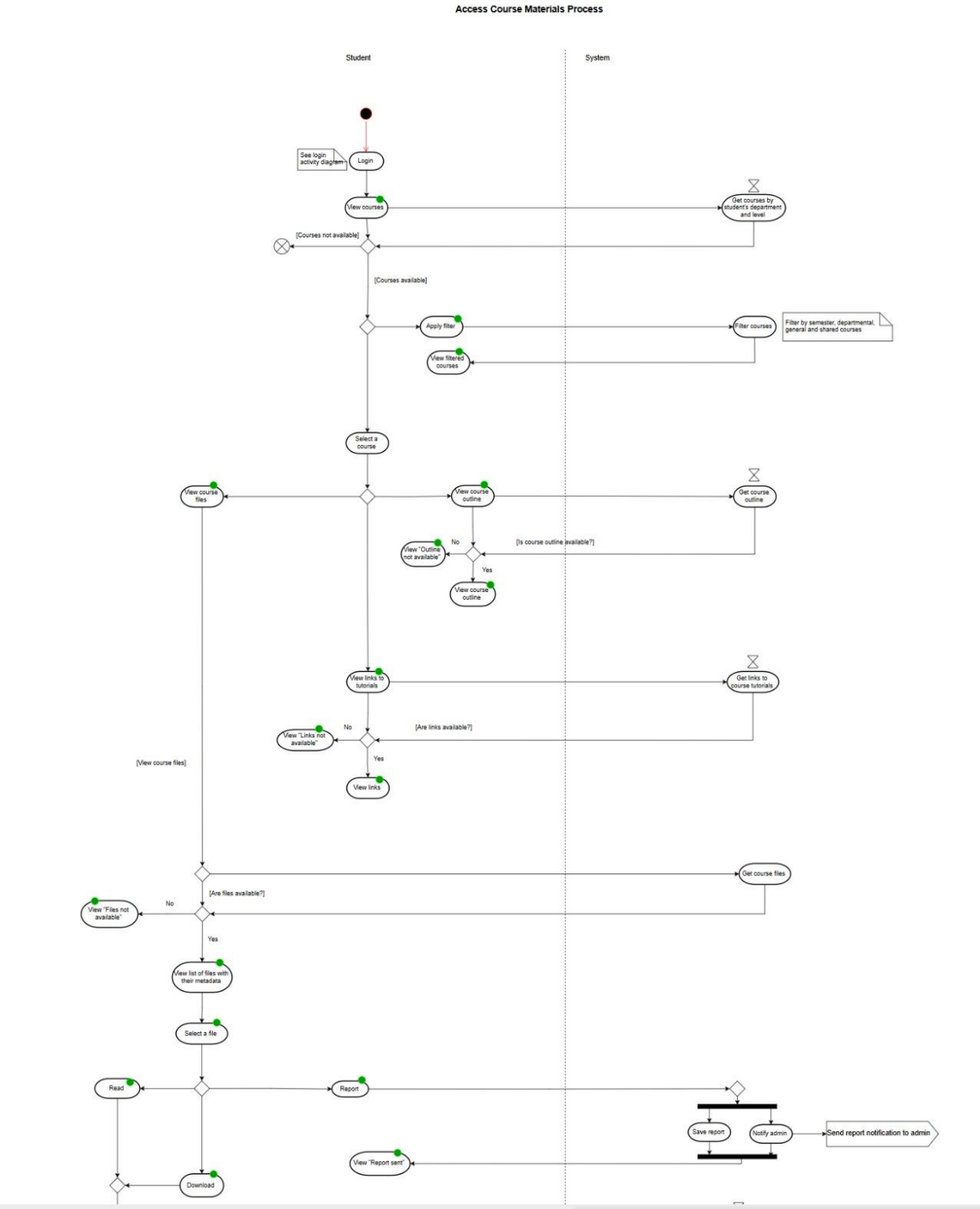


Figure 3.20 UML diagram showing access course material process

3.4 Development phase

The Development Phase marked the stage where all design plans, diagrams, and requirements were transformed into a fully functional web application. It involved translating the system architecture and user interface designs created in the Design Phase into executable code. During this phase, the team implemented both the frontend and backend components, integrated the API, configured the database, and connected the cloud storage system for managing study materials.

This phase also focused on ensuring security, functionality, and performance through proper authentication, data protection, and testing procedures. Collaboration tools such as GitHub and Docker were used to streamline version control, containerization, and deployment. By the end of the Development Phase, all major features of the UnibenEngVault platform were completed, tested, and prepared for deployment, making the system ready for real-world use by engineering students. The development phase passed through processes which include: Frontend, API, Backend, Database, Cloud Storage, Client vs Server, Programming languages, frameworks, and libraries. Applications. System secutity measures. Testing Phase

3.4.1Frontend

The frontend is the part of the application that students see and interact with directly. It takes the UI/UX designs and turns them into working pages. Think of it like a structural engineer applying an architect's drawings to build a house the frontend makes the visual plan functional.

In UnibenEngVault, the frontend is dynamic, meaning the information shown changes depending on the student using it. For example:

1. A student in the Electrical Engineering department sees courses and materials for their department and level only.
2. Personalized notifications, recent uploads, and relevant tutorials appear automatically.
3. Students can update their profile, change passwords, or view their course progress actions that trigger dynamic changes on the page.

There are two main types of frontend:

1. Static frontend: The content does not change and is the same for everyone, like a simple webpage with general information.

2. Dynamic frontend: Content adapts to the user, their actions, or the time. UnibenEngVault uses a dynamic frontend to make the platform personalized and interactive.

Why dynamic frontend matters: It improves the user experience by showing students relevant information only. It reduces confusion since students do not see irrelevant courses or materials.It allows real-time interactions, like searching for a file and seeing results immediately.

Frontend technologies we used: HTML & CSS**:** Provide the basic structure and styling for the pages. JavaScript: Makes the pages interactive, such as clicking buttons, updating content, and validating inputs. React: A JavaScript framework that allows us to create reusable components, manage dynamic content efficiently, and update the user interface without reloading the page.

By combining UI/UX design with dynamic frontend development, UnibenEngVault ensures that the platform is visually appealing, easy to use, and responsive to the needs of each student. figure 3.21 displays the frontend.

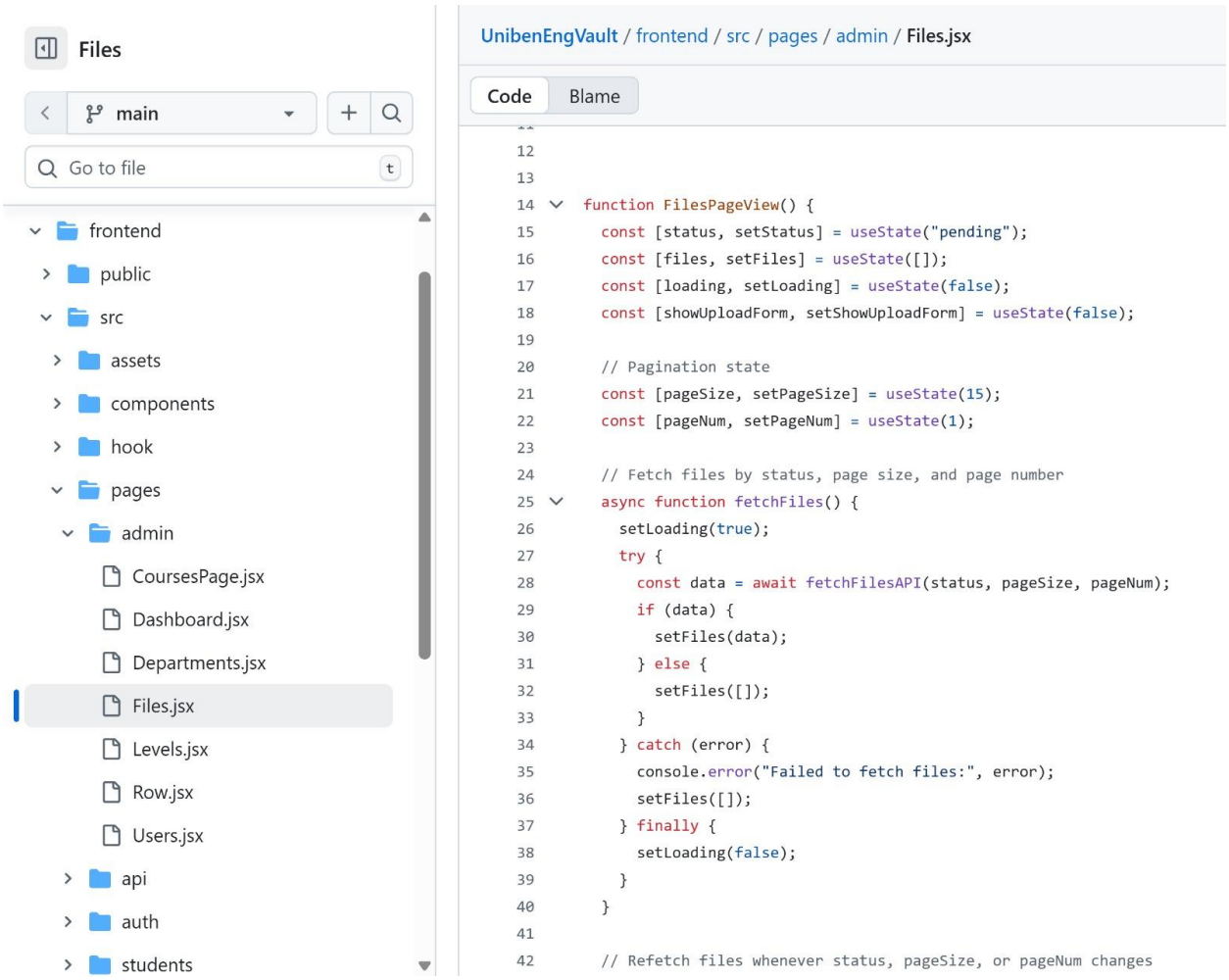


Figure 3.21 Frontend code for manage files page

3.4.2API (Application Programming Interface)

The API connects the frontend to the backend. It receives requests from the frontend, sends them to the backend for processing, and returns the responses back to the frontend. APIs are important because they:

1. Protect the backend: The API ensures that the frontend never directly accesses backend models or sensitive code. If an error occurs, the API handles it gracefully, sending only readable and safe messages to the frontend. This prevents hackers from learning about the system's internal workings.

2. Allow different programming languages to communicate: Normally, two programming languages might not work together due to syntax differences. The API solves this by using JSON (JavaScript Object Notation), a common data format that both frontend (JavaScript/React) and backend (Python/Flask) understand.

3. Implement security measures: The API can include authentication and authorization, ensuring only registered students and admins access the right data. It also validates incoming requests, filters out invalid or malicious input, and protects sensitive information like passwords.

In summary, the API acts as a middleman that not only connects the frontend and backend but also ensures secure, smooth, and reliable communication between them figure 3.22 shows a screenshot of the api system.

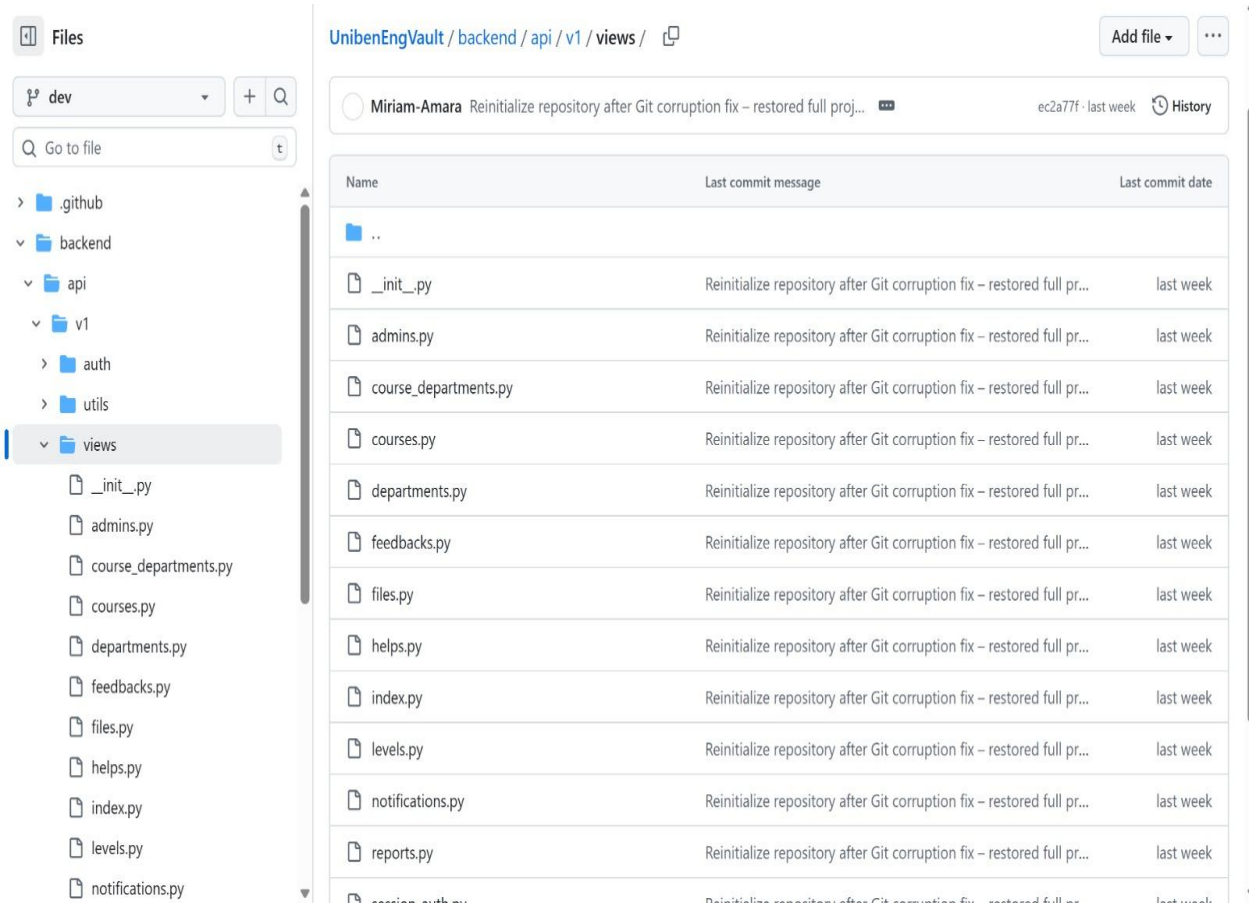


Figure 3.22 Backend Api python files

3.4.3Backend

The backend is the part of the application that students don't see, but it does most of the work. Think of it as the engine of a car invisible, but everything depends on it to run smoothly. In UnibenEngVault, the backend handles:

1. Data management: Storing and retrieving student details, courses, uploaded files, and tutorials.

2. Logic: Making sure students only see what belongs to their department and level and enforcing rules like who can upload or download files.

3. Performance: Handling multiple requests from students at the same time without slowing down.

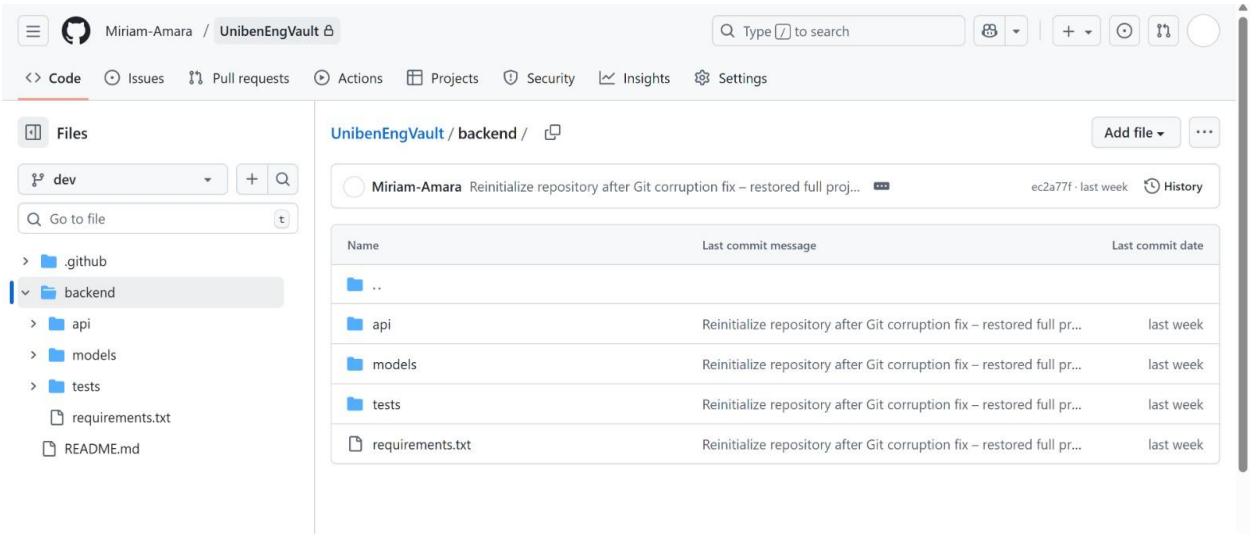


Figure 3.23 Folder structure for backend

3.4.4 How the backend works with the frontend:

When a student clicks a button, like “Search Past Questions,” the frontend sends a request to the API. The API passes this request to the backend, which queries the database, processes the data, and sends the response back through the API to the frontend. This ensures the frontend only receives the information it needs, in a secure and organized way.

Security in the backend:

1. Sensitive information like passwords is encrypted before being stored in the database.
2. Backend checks ensure that only valid and authorized requests get processed.
3. Combined with the API, the backend is never directly exposed to users, which prevents hackers from accessing internal code or data.
4. Error handling ensures that any issues are gracefully managed, showing users readable messages without exposing system details.

Technologies used for the backend:

1. Python: Chosen because the backend developer is most comfortable with it, making development faster and more reliable.
2. Flask: A lightweight web framework for Python that allows us to build the backend efficiently.
3. SQLAlchemy: A library that helps manage the database easily and securely.

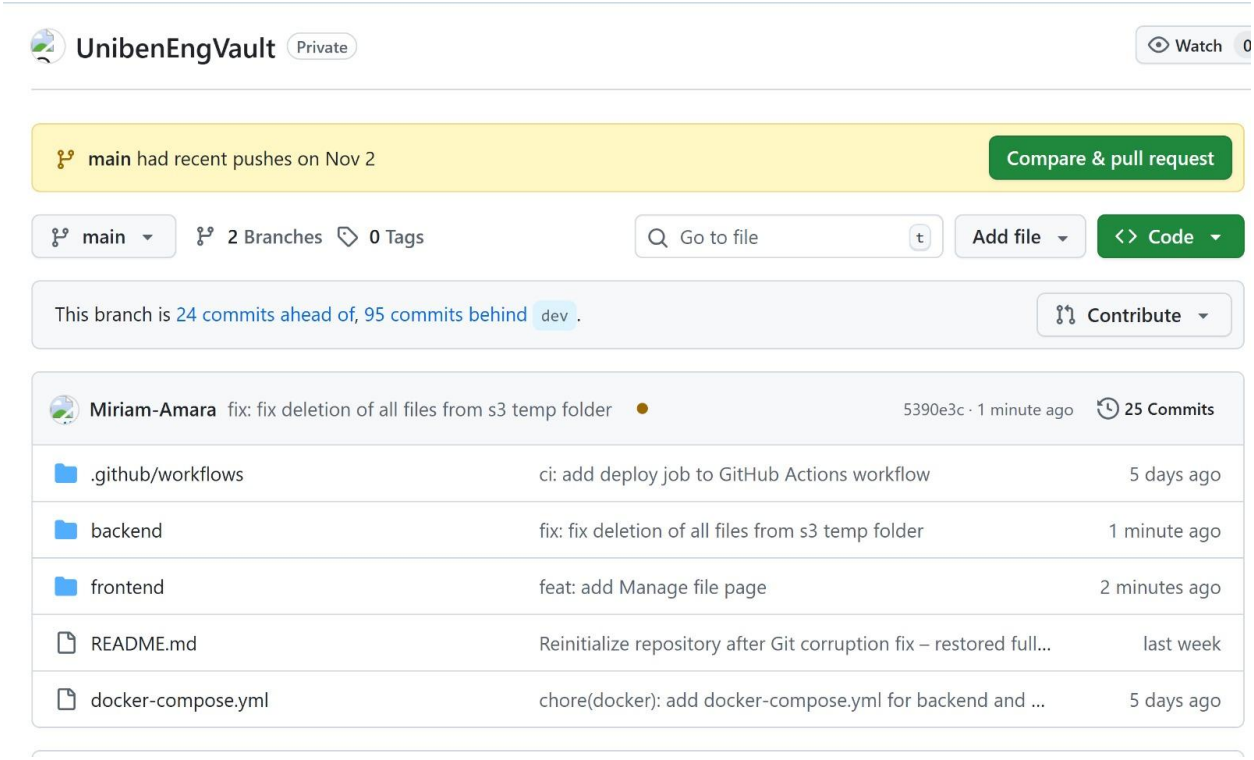


Figure 3.24 Shows the folder structure for UnibenEngVault showing both frontend and backend

3.4.5 Database

The database is where all structured information about students, courses, and files is stored. Think of it like a digital library where everything is organized, so the backend can find and deliver the right information to each student. Without the database, the system would not know which student belongs to which department or what courses they should access.

In UnibenEngVault, the database stores:

I. Student details like email, password, department, and level

II. Course information and associated study materials

III. Metadata about uploaded files such as type, upload date, and uploader

We used SQL (Structured Query Language) to manage the database. SQL is a standard language that lets us create, read, update, and delete data efficiently. It also allows us to define relationships between tables, like linking students to courses or files, which helps in building features like personalized dashboards and course access.

Security in the database:

1. Only authorized backend systems and APIs can access the database
2. Sensitive data like passwords is encrypted before storing
3. Access permissions ensure students cannot see files or data that are not meant for them
4. Regular backups prevent data loss in case of system failure

Technologies used for the database:

1. PostgreSQL (Postgres): A structured database system that uses SQL. Chosen because it is reliable, scalable, and easy to integrate with Python and Flask.

The database works closely with the backend and API. When a student requests a file, the backend queries the database using SQL, finds the correct record, and sends the information through the API to the frontend. This ensures secure and accurate delivery of data.

3.4.6 Cloud Storage

Cloud storage is used to store files that do not fit well in a structured database, like PDFs, past questions, and slides. It acts as a secure online vault where students can access study materials anytime and anywhere.

We used AWS S3 Bucket for cloud storage because:

1. It is reliable and keeps files available 24/7
2. It can easily scale if more files or students are added
3. It is secure, with strict access controls to prevent unauthorized downloads

Security in cloud storage:

1. Only authorized API requests can access files
2. Students can only access files for their department and level
3. Files are encrypted both while stored and during transfer

4. Access control rules prevent outsiders from downloading or viewing files they should not.

Together with the backend and API, cloud storage ensures that all study materials are safely stored, easy to access, and organized properly for students.

3.4.7Client vs Server

In web application development, the client and server represent two sides of a communication process that work together to deliver the user experience. Understanding their roles is crucial for designing efficient, secure, and scalable systems like UnibenEngVault.

The client refers to the user's device or browser for example, when a student opens the UnibenEngVault platform on their phone or laptop, their browser acts as the client. It is responsible for displaying the interface, capturing user actions (such as logins, searches, or downloads), and sending these requests to the server. The client runs the frontend part of the application, which was built using HTML, CSS, JavaScript, and ReactJS, and directly interacts with the student.

The server, on the other hand, is a powerful remote computer that hosts the web application, processes user requests, and manages the data. When the client sends a request for example, to download a course material the server receives it, checks permissions, retrieves the appropriate file or data from the database or cloud storage, and sends the response back to the client. The server handles the backend logic, security checks, and database communication using Python, Flask, and PostgreSQL.

This interaction follows a client server architecture, where each side performs specific roles:

1) Client Responsibilities:

a) Displaying the user interface and collecting user input.

b) Sending requests (e.g., login, search, download) to the server through the API.

c) Presenting responses, such as displaying course lists or downloaded files, in a clear and interactive way.

2) Server Responsibilities:

a) Receiving and validating client requests.

b) Processing business logic (e.g., verifying login credentials, checking access rights).

c) Fetching or updating information in the database.

d) Sending secure and structured responses (in JSON format) back to the client.

In UnibenEngVault, communication between the client and server happens through the API (Application Programming Interface), which acts as the bridge between both ends. The API ensures that only necessary information is exchanged and that all data transfers are secure and efficient. This separation of concerns allows the system to be more modular, meaning updates can be made to either the frontend (client) or backend (server) without affecting the other.

The client-server model also makes the system scalable and reliable. Multiple clients (students) can access the platform simultaneously, while the server manages all incoming requests efficiently using Gunicorn with multiple worker processes. This ensures fast response times even when the platform experiences high traffic.

In summary, the client and server operate in harmony to deliver the complete UnibenEngVault experience the client provides the interface and interactivity, while the server performs the heavy

lifting behind the scenes, ensuring secure data handling, efficient processing, and seamless communication between users and the system.

3.4.8 Programming Languages, Frameworks, Libraries, and Applications

During the design phase, we also discussed the main tools and technologies we would use to build UnibenEngVault. This helped everyone understand what each part of the system would use and why.

1 Frontend:

HTML & CSS: To structure and style the web pages. JavaScript: To make the pages interactive and respond to student actions. React: A JavaScript framework that allows us to build reusable components and manage dynamic content efficiently.

2 Backend:

Python: Chosen because the backend developer is most familiar with it, which makes development faster and more reliable. Flask: A lightweight Python framework used to handle web requests, API endpoints, and routing. SQLAlchemy: A library that simplifies database interactions and helps manage data securely.

3 Database:

SQL (Structured Query Language): Used to create, read, update, and delete structured data efficiently. PostgreSQL (Postgres): A reliable structured database system that works well with SQL, Python, and Flask.

3.4.9 Applications Used:

1. VSCode: For writing and managing code for frontend and backend.
2. Postman: For testing APIs and ensuring smooth communication between frontend and backend.
3. AWS S3 Bucket: For cloud storage of study materials.
4. Digital Ocean Droplet: For hosting and deploying the web application.
5. Git & GitHub: For collaboration, version control, and tracking code changes.

By choosing these languages, frameworks, and applications, we ensured that each part of the system could work together smoothly, the development process was efficient, and the platform remained secure, scalable, and maintainable.

3.4.10 Security Measures Implemented

To ensure that UnibenEngVault operates securely and protects all user information, several essential security measures were implemented during the development phase. These measures prevent unauthorized access, safeguard sensitive data, and maintain user trust.

1. Session Authentication

   The application uses session-based authentication to manage user logins securely. When a student logs in, a unique session token is created and stored on the server. This token identifies the user for a limited time, ensuring that only authenticated users can access protected pages or perform sensitive actions such as downloading materials. Sessions

automatically expire after a set period of inactivity to minimize the risk of unauthorized access.

2. Cross-Origin Resource Sharing (CORS)

   CORS was configured to control which external sources can interact with the platform's API. Only trusted domains are allowed, thereby preventing cross-site attacks such as Cross-Site Request Forgery (CSRF) or data hijacking. This adds an additional layer of protection between the frontend and backend, ensuring that requests originate only from verified sources.

3. Limited Data Collection (Privacy by Design)

   The platform was intentionally designed to avoid collecting private or unnecessary student information. Only minimal data such as name, department, and level is required for account creation and course personalization. No financial, biometric, or personally sensitive information is stored. This design choice aligns with ethical standards and modern data protection principles, ensuring that user privacy is preserved throughout all interactions.

4. Password Hashing with Bcrypt

   Passwords are never stored in plain text. Instead, the bcrypt algorithm is used to hash and salt passwords before saving them to the database. Even if the database were to be compromised, the original passwords cannot be retrieved. Bcrypt's adaptive nature also makes it resistant to brute-force attacks, providing a high level of password security for all users.

3.4.11 Testing Phase

Before deployment, comprehensive testing was conducted to verify that the system worked as intended and met all design, functional, and security requirements. Testing ensured stability, accuracy, and a positive user experience.

Development Testing

1. **Unit Testing**

   Unit tests were carried out to check individual components of the system. Each module such as user registration, file upload, or search functionality was tested independently to confirm that it produced the correct output. Unit testing helped the team identify and fix small issues early in the development process.

2. **Integration Testing**

   After unit testing, integration testing was performed to ensure that the different components (frontend, API, backend, and database) worked together seamlessly. For example, integration tests confirmed that when a student uploaded a file, the API correctly passed the data to the backend and stored the reference in the database. These tests also ensured that error messages and data validation were functioning properly across the system.

Both manual and automated testing methods were used during development. Automated testing was implemented using Python's unittest framework, while manual testing involved the development team verifying UI flows and API responses through tools such as Postman and browser-based inspection.

3.5 Deployment Phase

After successful testing and quality assurance, the project entered the deployment phase. The deployment process was carefully planned and executed to ensure reliability, scalability, and ease of maintenance.

1. **Docker Containerization**

   The entire web application was containerized using Docker, which packaged all dependencies, configurations, and code into a lightweight, portable unit. This ensured that the application behaved consistently across development, testing, and production environments.

2. **Continuous Integration and Deployment (CI/CD) with GitHub Actions**

   A CI/CD pipeline was set up using GitHub Actions. Each time the team pushed new updates to the repository, automated scripts ran tests, built the Docker image, and deployed the updated version. This approach reduced manual work, eliminated deployment errors, and ensured faster delivery of new features.

3. **Hosting with Digital Ocean Droplet**

   The production environment was hosted on a Digital Ocean Droplet, a virtual private server known for its reliability and scalability. The droplet provided a stable hosting infrastructure, allowing the platform to handle multiple simultaneous users efficiently.

4. **Gunicorn Application Server (3 Workers)**

   The backend was served using Gunicorn, a high-performance WSGI server for Python

applications. Three workers were configured to handle concurrent requests, improving speed, responsiveness, and overall system performance.

5. **Domain Name and HTTPS Encryption**

   A custom domain name was registered and linked to the platform to enhance accessibility and professionalism. To ensure secure data transmission, an SSL certificate was installed, enabling HTTPS communication. This encrypted all traffic between users and the server, protecting login credentials and other sensitive interactions.

6. **User Testing and Feedback**

   After deployment, user testing was conducted with a group of engineering students from different departments. The goal was to observe real-world usage, identify usability challenges, and collect feedback on performance and design. Based on the feedback received, several improvements were made to enhance responsiveness on mobile devices, adjust text readability, and optimize loading speed.

3.5.1 Post-Deployment Monitoring and Maintenance

Even after deployment, continuous monitoring and maintenance were carried out to ensure smooth operation and system reliability.

1. Error tracking tools were configured to monitor performance and automatically log issues.
2. Server uptime monitoring was set up to ensure that the platform remained accessible at all times.
3. Regular updates and security patches were applied to keep the platform secure from newly discovered vulnerabilities.

4. User feedback was continually reviewed to guide further feature improvements and UI refinements.

Through these steps, the UnibenEngVault system achieved a stable, secure, and user-friendly deployment, ensuring that students could confidently access materials and engage with the platform without technical interruptions.

CHAPTER FOUR

RESULTS AND DISCUSSION

4.0 Introduction

This chapter presents the implementation of the UnibenEngVault web application and the results obtained after the system was developed and deployed. It describes how the various modules designed in Chapter Three were integrated into a fully functional platform, explains the technologies and environments used during implementation, and highlights the outcomes of user testing, performance evaluation, and system functionality verification.

The implementation process focused on turning the planned architecture, designs, and workflows into a working system accessible to engineering students. This chapter also presents screenshots of the developed interfaces, user interactions, and system responses to demonstrate how effectively the platform meets its stated objectives.

4.1 System Implementation Environment

The UnibenEngVault web application was implemented using a combination of modern technologies to ensure performance, reliability, and scalability. The implementation environment consisted of the following:

1. **Frontend Technologies:**

   HTML, CSS, JavaScript, and ReactJS were used to develop the user interface and dynamic content of the platform. React components made the pages interactive and reusable, while CSS was used for styling to match the UI/UX design created in Figma.

2. **Backend Technologies:**

   Python and Flask formed the core of the backend. Flask handled API routing, authentication, and logic for data management, while SQLAlchemy was used for Object Relational Mapping (ORM) to interact with the database securely.

3. **Database:**

   PostgreSQL served as the structured database system. It stored all user data, course details, file metadata, and other essential information.

4. **Cloud Storage:**

   AWS S3 Bucket was used to store large files such as lecture slides, past questions, and notes. It ensured high availability, security, and easy access for users.

5. **Server Environment:**

   The application was deployed on a DigitalOcean Droplet, running on Ubuntu Linux. Gunicorn (with three workers) served as the application server, and Nginx acted as the reverse proxy.

6. **Development Tools:**

   Visual Studio Code (VSCode) for code development. Git and GitHub for version control and collaboration. Postman for API testing. Docker for containerization and deployment

4.2 System Implementation Process

The implementation process followed a modular development approach, where each system component was developed, tested, and integrated in stages.

1. **Frontend Implementation**

   The frontend was implemented using ReactJS, based on the UI/UX design prototypes from Figma. Each page such as the homepage, login page, file page, and admin dashboard was built as a separate component. React routing was configured to ensure smooth page transitions without reloading.

   Responsiveness was achieved using CSS Flexbox and Grid, allowing the layout to adjust automatically to different screen sizes.

2. **Backend Implementation**

   The backend was implemented using Flask, which handled requests from the frontend and communicated with the database through SQLAlchemy. Routes were created for actions such as user registration, login, file uploads, and downloads. Flask's blueprint structure helped separate student, admin, and API routes for better organization.

3. **Database Implementation**

   The database schema was implemented in PostgreSQL based on the ERD designed in Chapter Three. Tables were created for students, courses, files, departments, and tutorials, each linked through foreign keys. Sample test data was inserted to verify relationships and queries.

4. **API Integration**

   After both frontend and backend were implemented, the RESTful API endpoints were integrated. Endpoints allowed secure communication between the interface and database. JSON was used for data exchange to ensure compatibility between the React frontend and Flask backend.

5. **Cloud Storage Setup**

   AWS S3 was configured to store and retrieve uploaded materials. Access permissions were restricted through API tokens to ensure that only authorized users could upload or download files.

6. **Deployment Implementation**

   Once all modules were tested and stable, the application was containerized using Docker and deployed on the DigitalOcean Droplet. Continuous integration with GitHub Actions automated the deployment process whenever new code was merged into the main branch.

4.3 System Features

The final system includes several features that directly address the challenges identified in the research phase:

1. **User Registration and Login:**

   Allows students to create an account and access materials based on their department and level.

2. **Course Material Access:**

   Students can easily search, view, and download past questions, slides, and other study resources.

3. **Admin Dashboard:**

   Provides administrators with tools to manage users, departments, courses, and uploaded materials.

4. **Notifications and Updates:**

   Displays new uploads, tutorial links, and announcements relevant to each department.

5. **Responsive Design:**

   The application adapts to mobile, tablet, and desktop screen sizes for smooth use on any device.

4.4 System Testing and Evaluation

After deployment, both technical testing and user testing were conducted to ensure that the platform met functional and usability standards.

1. **Functional Testing:**

   Each module was tested to confirm that it performed its expected functions correctly. For example: Login and registration authentication was verified with valid and invalid credentials. File uploads and downloads were tested for speed and accuracy. Admin functionalities (such as course and file management) were validated for permission accuracy.

2. **Performance Testing:**

   The system was tested under simulated user loads to measure response time and reliability. The use of Gunicorn and Docker ensured stable performance even when multiple users accessed the platform simultaneously.

3. **User Testing:**

   Selected engineering students were invited to test the platform. They were asked to register, search for courses, and download files while providing feedback on ease of use,

navigation, and visual design.

The feedback revealed that most users found the platform intuitive and helpful, with particular appreciation for the organized layout and fast download features. Minor UI adjustments were made based on their input.

4. **Security Testing:**

   The platform was tested for vulnerabilities such as unauthorized access, data exposure, and session hijacking. The use of bcrypt password hashing, session authentication, and HTTPS encryption successfully prevented such attacks during testing.

4.5 Findings

The implementation and testing phases yielded the following results:

1. The current level of access to educational materials and digital tools within the Faculty of Engineering is low.
2. Information and data gotten from investigation was analyzed and documented.
3. All lecture materials, textbooks and practical resources were collected, organized and digitalized.
4. User story research was conducted and user flow charts were created for the web app.
5. A web based course material management system for all departments within the faculty was designed and developed.
6. The app was deployed and tested for all departments within the faculty of engineering

7. The effectiveness of the system was monitored to ensure it aided in the improvement of access to academic resources and overall student learning outcome.

These results demonstrated that the platform not only addressed the initial problem but also achieved the project's goals of accessibility, security, and usability.

4.6 Summary

This chapter discussed the implementation and testing of the UnibenEngVault web application. It explained the technologies used, the process of integrating different modules, and the testing conducted to ensure performance, functionality, and security. The results show that the system successfully provides a centralized and reliable platform for engineering students to access study materials conveniently.

CHAPTER FIVE

CONCLUSION, AND RECOMMENDATIONS

5.1 Conclusion

This chapter presents the summary, conclusion, and recommendations drawn from the development of the UnibenEngVault web application. It provides a concise overview of the work carried out from problem identification to system deployment and highlights key achievements, lessons learned, and suggested improvements for future work. The UnibenEngVault project was developed to address a major challenge faced by engineering students at the University of Benin the lack of a centralized and easily accessible platform for sharing and retrieving academic materials such as past questions, lecture slides, and tutorial notes.

The project began with a detailed planning and research process that involved identifying the problem, gathering user requirements through questionnaires, and analyzing feedback from over 600 engineering students. This research confirmed the need for a platform that is accessible, organized, and user-friendly.

The Design Phase translated these findings into system models, wireframes, and UI/UX layouts. Using Figma, the team created interactive prototypes that served as the visual guide for development. UML and ERD diagrams were also created to define system structure and database relationships, ensuring logical and consistent system behavior.

During the Development Phase, the frontend was built using ReactJS, while the backend was developed with Python (Flask), and PostgreSQL served as the main database. Cloud storage was implemented using AWS S3, and all components were integrated through secure RESTful APIs.

Security measures such as session authentication, CORS configuration, and password hashing with bcrypt were implemented to ensure data protection and system integrity. The application was containerized with Docker, deployed using DigitalOcean Droplets, and managed with CI/CD pipelines through GitHub Actions.

Comprehensive testing was conducted, including unit and integration tests, as well as user testing with selected students. Feedback from users confirmed that the platform was intuitive, responsive, and valuable for their academic needs. The final deployed version met the core objectives of accessibility, security, and ease of use. The development of the UnibenEngVault web application successfully achieved its primary objective to create a centralized, secure, and easy-to-use digital repository for engineering students. The system provides an efficient means for students to access study materials and allows administrators to manage files and user data seamlessly.

The integration of modern technologies such as ReactJS, Flask, PostgreSQL, Docker, and AWS S3 ensured a robust, scalable, and high-performing platform. The attention to UI/UX design also contributed significantly to the positive user experience, making the platform both functional and visually appealing.

Overall, the project demonstrated the importance of combining technical proficiency, teamwork, and user-centered design in software development. It also emphasized how engineering students can apply theoretical knowledge to solve real-life challenges within their academic community.

5.2 Recommendations

While the current version of UnibenEngVault performs effectively, several enhancements can be considered for future development to improve functionality, scalability, and user experience:

1. **Mobile Application Version:**

   Developing a mobile app version for Android and iOS would improve accessibility and convenience for students who rely primarily on smartphones.

2. **Offline Access:**

   Enabling offline file caching or downloads within the app would allow students to study materials even without an active internet connection.

3. **Analytics and Usage Tracking:**

   Introducing an analytics dashboard for administrators to track user activity, file downloads, and popular courses would help improve content management and identify student needs.

4. **Enhanced Security Features:**

   Future updates could include two-factor authentication (2FA) and regular automated security audits to strengthen data protection and user privacy.

5.3 Findings of the study

1. Successfully designed and implemented a secure, centralized learning platform for engineering students.

2. Conducted extensive user research and integrated user feedback into design and testing.

3. Applied modern technologies (ReactJS, Flask, PostgreSQL, Docker, AWS S3) for performance and scalability.
4. Deployed the application using CI/CD automation for efficient updates and maintenance.
5. Ensured data security and privacy through encryption, authentication, and limited data collection.
6. Achieved a functional, responsive, and user-friendly interface accessible on multiple devices.

REFRENCES

ALX Africa. (2024). *Software Engineering Program.* Retrieved from https://www.alxafrica.com (Provided foundational knowledge on version control, deployment, and collaboration tools.)

Amazon Web Services (AWS). (2024). *Amazon S3 User Guide.* Retrieved from https://aws.amazon.com/s3 (Referenced for cloud storage setup and access management.)

Coursera. (2023). *Full Stack Web Development Specialization.* Retrieved from https://www.coursera.org (Used to gain understanding of frontend, backend, and API development processes.)

DigitalOcean. (2024). *Droplets and App Deployment Documentation.* Retrieved from https://docs.digitalocean.com (Referenced for server setup, domain configuration, and HTTPS deployment.)

Docker, Inc.. (2024). *Docker Overview.* Retrieved from https://docs.docker.com (Used for application containerization and deployment process.)

Figma. (2024). *Design and Prototyping for Teams.* Retrieved from https://www.figma.com (Used as the main UI/UX design tool for creating prototypes and interactive layouts.)

GitHub. (2024). *CI/CD with GitHub Actions.* Retrieved from https://docs.github.com (Used to configure continuous integration and automated deployment workflows.)

Mozilla Developer Network (MDN). (2024). *HTML, CSS, and JavaScript Documentation.* Retrieved from https://developer.mozilla.org (Referenced for frontend development, DOM manipulation, and web standards.)

Nielsen Norman Group. (2022). *10 Usability Heuristics for User Interface Design.* Retrieved from https://www.nngroup.com (Used to guide user experience (UX) design principles and interface evaluation.)

Pinterest. (2024). *UI/UX Design Inspiration and Trends.* Retrieved from https://www.pinterest.com (Used as a creative reference for color schemes, layouts, and modern design ideas.)

PostgreSQL Global Development Group. (2024). *Official PostgreSQL Database Guide.* Retrieved from https://www.postgresql.org/docs (Used for database schema design, querying, and data management.)

React. (2024). *Official React Guide.* Retrieved from https://react.dev (Used for building the dynamic frontend and managing component-based architecture.)

Pallets Projects. (2024). *Flask Documentation: The Python Microframework for Web Development.* Retrieved from https://flask.palletsprojects.com (Referenced for backend API development and server configuration.)

W3Schools. (2023). *Web Technologies Tutorials.* Retrieved from https://www.w3schools.com (Referenced for additional guidance on frontend scripting and styling practices.)

APPENDIX

STUDENT SURVEY – COURSE MATERIAL WEB PLATFORM

This short survey will help us understand how you currently get your course materials and what features you'd like us to include in the web platform we're building.

It won't take more than 3–5 minutes to fill.

Thanks so much for your time!

A. Student Information

Are you an Engineering student?

☐ Yes

☐ No

Department: _____

Level (e.g., 100L, 200L): _____

B. Section 1: General Opinion

a. Do you like the idea of a website where you can view and download course materials and past questions?

☐ Yes

☐ No

☐ Not sure

b. Would it help you be more prepared for exams?

☐ Yes

☐ No

☐ Not sure

c. Would you recommend the web platform to your friends?

☐ Yes

☐ No

☐ Maybe

d. What problems do you face when getting academic materials? (Check all that apply)

☐ Outdated materials and past questions

☐ Don't know where to find materials

☐ Takes too long to receive materials from seniors and colleagues

e. Have you ever had a bad grade because you couldn't find the materials you needed?

☐ Yes

☐ No

f. What kind of materials do you need most? (Choose all that apply)

☐ Notes

☐ Past questions

☐ Assignments

☐ Textbooks

☐ Project samples

☐ Videos

☐ Lecture materials

g. Do you usually save or download materials for later use?

☐ Yes

☐ No

h. Do you find it difficult to track what you've read?

☐ Yes

☐ No

i. What features would you like the website to have? (Check all that apply)

☐ Search bar

☐ Materials organized by department/level

☐ Upload by students

☐ Course previews

☐ Course outline

☐ Course ratings

☐ Track what you've studied

☐ Upload your answers to past questions

☐ See others' answers to past questions

☐ Links to YouTube tutorials and other useful resources