**CITC - IT Department**

Computer Laboratory Activity Form

| Course Code | | IT323 | |
| --- | --- | --- | --- |
| Course Title | | Application Development and Emerging Technology | |
| Topics Covered | | Overview of Software and Hardware Technology | |
| Objectives | | The objective of this comprehensive laboratory activity is to provide students of IT323 with an in-depth understanding of cross-platform application development by integrating mobile, web, and emerging technologies. Students will gain practical experience in building a feature-rich application that leverages the capabilities of both mobile and web platforms, while also exploring cutting-edge technologies for enhanced functionality and user experience. | |
| Materials:   1. Computers with internet access 2. Development environments for web and mobile platforms (Visual Studio Code, and Android Studio) 3. Visual Studio Code, and Android Studio 4. Smartphone and Android Studio Emulator 5. Internet connectivity 6. Database system (MySQL) 7. Duration: 1 month | | | |
| **Description**  **Title :** JudgeMentor: Your Path to Success  **Activity Steps:**   1. Introduction to Cross-Platform Development and Emerging Technologies:   JudgeMentor is a versatile cross-platform system designed to facilitate the creation of reviewers for upcoming exams and quizzes. This innovative platform offers a wide array of benefits and holds significant importance in modern software development. Here is an overview highlighting the system's importance and advantages in the context of modern software development.  Cross-Platform Compatibility  JudgeMentor is built to run seamlessly across different platforms such as web, iOS, and Android. This cross-platform compatibility ensures that users can access the system from various devices, enhancing flexibility and accessibility. Developers can reach a broader audience and provide a consistent user experience across different platforms.  Efficient Reviewer Creation  JudgeMentor streamlines the process of creating reviewers for exams and quizzes, enabling users to generate customized review materials quickly and efficiently. This feature is particularly beneficial for developers and educators who need to create assessment content for software development courses, coding challenges, or technical interviews.  Personalized Learning Paths  JudgeMentor offers personalized learning paths tailored to individual users' needs and preferences. Developers can track their progress, identify areas for improvement, and focus on specific topics relevant to their learning goals. This personalized approach enhances the effectiveness of learning and skill development in software development.  Cost-Effective Learning Solution  By offering a comprehensive set of features and functionalities, JudgeMentor provides a cost effective learning solution for developers seeking to enhance their skills and knowledge in software development. The system offers a valuable resource for self-paced learning, exam preparation, and skills assessment without requiring significant financial investment.   1. Setting Up Development Environments:   Node.js and npm Download and install Node.js from the official website. Verify the installation by running node -v and npm -v commands in the terminal or command prompt. Use npm to install global packages such as React Native CLI or Flutter SDK if using npm-based frameworks.  IDE (Integrated Development Environment)  Download and install an IDE or code editor like Visual Studio Code. Install relevant extensions for cross-platform development, such as React Native Tools or Flutter/Dart extensions. Configure settings and preferences according to personal or team preferences.  Cross-Platform Development Framework  Install the chosen framework's SDK or CLI tools. For example, install React Native CLI via npm or download Flutter SDK from the official website. Follow the framework's documentation to set up the development environment for each target platform (Android, iOS, web).  Version Control System  Install Git from the official website. Set up a Git repository for the "JudgeMentor" project locally or on a hosting platform like GitHub. Configure Git settings such as name and email using git config.  Dependency Management  Initialize a new npm project (npm init) or Flutter project (flutter create) in the project directory. Use npm to install project dependencies listed in the package.json file or Flutter's pubspec.yaml file. Ensure that dependencies are properly installed and listed in the respective lock files (package-lock.json or pubspec.lock).  Testing Frameworks  Install testing frameworks and libraries using npm or Flutter's pub package manager. Write unit tests, integration tests, and UI tests using the chosen frameworks (e.g., Jest, Mocha, XCTest, Flutter's test package). Run tests locally to ensure they pass before committing changes to the repository.  **Software Application Development Methodology**  **Waterfall Model**  In utilizing the waterfall methodology, the development of cross-platform " JudgeMentor " progresses through distinct, sequential phases, adhering to a structured approach. Beginning with requirements gathering and analysis, followed by system design, implementation, testing, and deployment, each stage is meticulously planned and executed before proceeding to the next. This method ensures thoroughness and clarity in every aspect of " JudgeMentor " development, from its user interface design to its backend functionality.    **Figure 1.1 Waterfall Model Diagram**  **Requirements Gathering**  Initially, stakeholders' input regarding JudgeMentor's functionalities, usability, and target audience is gathered through surveys, interviews, and market research for the mobile and web requirements. Comprehensive documentation is prepared, outlining the desired features, platform compatibility, and user expectations for both mobile and web. The requirements are refined iteratively based on feedback to ensure clarity and alignment with the users’objectives.  **System Design**  Mobile Design   * Design the system architecture for the mobile application, including user interface layout, navigation flow, and data storage structure. Create wireframes and mockups to visualize the UI/UX design for different screens and user interactions. * Database design, create tables for users, questions, reviewer, and etc.   **Web Design**   * Design the architecture and user interface for the web application, ensuring responsiveness and usability across different devices and screen sizes. Create mockups and prototypes to visualize the layout and user flow of the web * Database design, create tables for users, questions, reviewer, and etc.   **Implementation**  Mobile Implementation   * Choose appropriate technologies and frameworks for mobile development (e.g., React Native, Flutter). * Application can run and used in both android and iOS. * User can use the application with a minimum of 4gb on device’s storage. * Implement user authentication and authorization. * Develop modules for exam selection, review materials, quizzes, and progress tracking. * Integrate with external APIs for accessing study materials or exam content. Implement offline mode functionality for accessing materials without internet connectivity.   Web Implementation   * Choose appropriate technologies and frameworks for web development (e.g., React.js, Angular, Vue.js). * Implement user authentication and authorization. Develop web pages for exam selection, review materials, quizzes, and progress tracking. Ensure responsiveness for various screen sizes and browsers. Integrate with external APIs for accessing study materials or exam content. * System can run in windows 10   **Verification (Testing)**  Mobile Testing   * Allow users to test the mobile application to determine if there are bugs that needs to be fixed. * The administrator must also test the application by actively using the app to know if errors and bugs occurs. * Conduct unit testing for each web component to ensure functionality. Perform integration testing to verify interactions between different web pages. Conduct cross-browser and cross-device testing to ensure compatibility.   Web Testing   * Allow users to test web system to determine if there are bugs which needs to be fixed. * The administrator must also test the system by actively using the system to know if errors and bugs occur. * Conduct unit testing for each web component to ensure functionality. * Perform integration testing to verify interactions between different web pages. * Conduct cross-browser and cross-device testing to ensure compatibility.   **Verification (Deployment)**  Mobile Deployment   * Ensure that the "JudgeMentor" mobile application is fully developed and tested. * Generate a signed APK (Android Package) file for deployment. This involves configuring signing keys and generating a signed APK using Android Studio or the command line. * Create a release version of the APK with appropriate versioning and release notes. * Optimize the APK size by removing * Log in to the Google Play Developer Console. * Create a new application entry for "JudgeMentor" if not already done. * Upload the signed APK file to the Google Play Developer Console. * Fill in the required details, including title, description, screenshots, and pricing. * Publish the application to the Google Play Store.   Web Deployment   * Select a web hosting provider for deploying the "JudgeMentor" web application. Options include AWS, Google Cloud Platform, Microsoft Azure, or specialized hosting providers like Netlify or Vercel. * Log in to the chosen hosting provider's dashboard. * Set up a new project or website deployment. * Upload the application files to the hosting provider's server using FTP, SSH, or a built-in deployment tool. * Configure DNS settings to point the domain name to the deployed application.   **Maintenance and Support**  Maintenance and Support for Mobile App  Bug Fixing and Troubleshooting - Continuously monitor user feedback and error reports to identify and prioritize bugs. Develop patches and updates to fix reported issues promptly. Conduct regression testing to ensure that bug fixes do not introduce new issues.  Performance Optimization - Monitor app performance metrics such as loading times, responsiveness, and resource usage. Identify areas for optimization, such as reducing memory footprint, improving UI responsiveness, and optimizing network requests. Implement performance enhancements through code refactoring and optimization techniques.  Compatibility Updates - Stay up-to-date with changes in mobile operating systems (iOS, Android) and third-party libraries/frameworks. Ensure compatibility with new OS versions and device models by updating dependencies and adjusting code as necessary. Test the app on a variety of devices and OS versions to verify compatibility.  Feature Enhancements - Gather feedback from users and stakeholders to identify potential new features or improvements. Prioritize feature requests based on user needs and market trends. Develop and deploy new features through iterative updates to the mobile app.  Maintenance and Support for Web Platform  Bug Fixing and Issue Resolution - Monitor error logs and user feedback to identify and prioritize bugs and issues. Develop and deploy fixes for reported issues in a timely manner. Conduct thorough testing to ensure that bug fixes are effective and do not introduce new problems.  Performance Monitoring and Optimization - Monitor website performance metrics such as page load times, server response times, and resource utilization. Identify performance bottlenecks and areas for optimization, such as database queries, server-side processing, and frontend rendering. Implement optimizations to improve website speed and responsiveness.  Browser Compatibility and Responsive Design - Regularly test the website across different web browsers and devices to ensure compatibility and consistent user experience. Address any layout or functionality issues that arise due to browser differences or screen sizes. Maintain responsive design principles to ensure that the website displays correctly on various devices, including desktops, tablets, and smartphones.  Content Updates and Management - Regularly update website content, including study materials, exam information, and other relevant resources. Implement content management tools and workflows to streamline the content update process. Ensure accuracy and relevance of information presented on the website.   1. Ideation and Project Planning:     **Figure 1.2 Context Diagram Level 0 of JudgeMentor for Mobile and Web**    The context diagram depicted in Figure 1.2 above explains the functionality of the system as a whole. As what is shown in the diagram, the admin provides maintenance to the system and all registered users are forwarded to the admin. The user manages the reviewer, can add, update, and delete reviewers to the system. The project manager then provides information, manage and monitor projects to the JudgeMentor reviewer system.      **Figure 1.3 Context Diagram Level 1 for JudgeMentor Mobile and Web**  Data flow diagram level 1 provides a detailed representation and visualization of the system’s processes. In registration process, user request for registration where credentials are stored in the user credentials database. It is the admin who confirms the registration of the user. Next, in login process, user provides login details which is also stored in the same database while the login process confirms user login into the system. Lastly, user can manage reviewer as well as add, update, and delete reviewer. Reviewer information are stored in the user reviewer details.    **Figure 1.4 Child Diagram Level 1 for JudgeMentor Mobile and Web**  The figure 1.4 above shows the context diagram level 1 but exploded. First the registration request is done by the user on both mobile and web. Next, the login is done by the user on both mobile and web as well. Lastly, the user can manage reviewers on both mobile and web.    **Figure 1.5 Use Case Diagram for JudgeMentor Mobile**    **Figure 1.6 Use Case Diagram for JudgeMentor Web**  As depicted in figure 1.5 and 1.6, it shows the interactions that takes place in the system. This illustrates the relationship and how the user, admin, and project manager interact with the system. First, the user login using the user’s credential while the admin confirms the login process and will send notification to the user once successful. The user can then manage the reviewer where user can add, update, and delete reviewers that were added and stored in the system. The user can also mark as done a reviewer once done as a part of managing its projects.       4. Designing the User Interface:  https://lh7-us.googleusercontent.com/0CAkgwqZ44479O_4H6ul_uJZshZhpWco3GWqGM-ZPO0E06HJ7X7UUbjH0ImJOek_UACIkj0a7iDBWBXpRr6sVurxcVd2rR6qNdBgtZYBT49dfub5EuzByVxQBVbXGHGkfpBrCc48e0ucTj9_dB7S6IEhttps://lh7-us.googleusercontent.com/5UtT51yS0eTaVo8wMvz4vdnuOaKlYfVaoKFiHp-AA41V12XinxPV3M9pxxDEYe9YpirvPG-eyX7viLhqVveuRVrlHuEE9SmlbxPRl_UGtVHNLYHeEz078zL6QyCkgWbWE9rxy81X6v14Xs0xay027w0    **Figure 1.1 Sign up Page (Web)                               Figure 1.2 Login Page (Web)**  https://lh7-us.googleusercontent.com/YryTQPoyTW54D0HnfLvIOHEomT4KuXYASO224oB6cnQHEPIGoyqi9-U_A-BnMTX7B_WOIgluSCKVpG_6l6xzxx-ffJEvDN5kqkG3ngidfpEnb66cJnzxpq_npxIfY00TUkttA64ovsdVIrd1pqzdbkAhttps://lh7-us.googleusercontent.com/CK1W_lgjA0GlsERFagOffSbi4Rrfol2l_YsgOCBgkEVvDMfUHn0gDFwun7A6oi2tXsbuqdBp1vMBeByD-dflM0r0ipwVprPIHr63OUfeK5AapTGm3we8G0xFjpEEbQX3jCgMZKkOta3iOJrLvTKJMII    **Figure 1.3 Home Page (Web)                              Figure 1.4 Reviewer Page (Web)**  https://lh7-us.googleusercontent.com/l75XDYfcu94C20NHCtcxK9JCYXsDL23gHcpUYeNkCeKEPETW5WIOOUczjl-sp69wec4uNgUhBzzM7PHROA7CuiugH7KYtmciF6ICuArDPoWnYeqgFnBuBDch14ggIlxFlvZFFQeMtqZk0EFGuWe_wr0https://lh7-us.googleusercontent.com/L7_PpTnpMAsukEtNtmTTEDunboHvlMqXBwt-Mg-b9OvSVtkq7hdjPNlR03hlEyA2RnvV6qcEZQ2m-hXDHLqA1Z0ra05tBLUn-f-h00peyThNWplWiPsFD7urJXUGjMVjnkyktiysCAc9YziGmd9-CXIhttps://lh7-us.googleusercontent.com/io2DKMm3n7uDAs7bFb_Xzb1Qbx1q3MOSuFVaFPoJKrejtbLYa2wIaohcbX3XS4Xg8wcv0NjZNkD7C6X3CIbenQz5ceJy_b-rnlB1byBHv6UJ4K2qY7KNqTz66jW1KIoC6X78i2Ja8Z3vGpnQv8NBHjE    **Figure 1.5 Sign Up                       Figure 1.6 Login                   Figure 1.7 Viewing Page**     5. Implementing the Web Interface:  Implementing the web interface of the JudgeMentor application is a critical phase of development. By utilizing HTML, CSS, and JavaScript, the group ensures the platform is visually appealing, user-friendly, and functional. The group emphasizes responsive design to guarantee the application looks and functions seamlessly across various devices and screen sizes. Employing responsive design techniques such as flexible grids, fluid layouts, and scalable images, the group ensures optimal viewing and interaction experiences. Additionally, cross-browser compatibility is prioritized. Thorough testing on different web browsers such as Google Chrome, Mozilla Firefox, Safari, and Microsoft Edge ensures consistent performance and appearance. By addressing any compatibility issues, the group guarantees JudgeMentor performs optimally regardless of the browser being used. This emphasis on responsive design and cross-browser compatibility ensures that JudgeMentor delivers a seamless and enjoyable user experience to all users.  When implementing the web interface for JudgeMentor: Your Path to Success, it's crucial to prioritize responsive design and cross-browser compatibility to ensure a seamless user experience across various devices and browsers.  **HTML Structure:**  To structure a web interface with HTML, begin by declaring the document type as HTML5. Then, define the language in the `<html>` tag. In the `<head>` section, set metadata like character encoding and viewport settings, and specify the title of the webpage. Link your CSS file for styling. In the `<body>` tag, organize content with semantic elements like `<header>`, `<main>`, `<footer>`, and `<section>`. Use `<nav>` for navigation links, usually placed in the `<header>`. The main content goes in `<main>`, and any footer content goes in `<footer>`. This concise approach ensures a structured and manageable HTML codebase for your web interface.  **Styling with CSS:**  When styling with CSS, use selectors and properties to apply styles to the HTML structure, employ classes and IDs for targeted styling within the HTML elements, arrange layout with positioning or flexbox within the HTML layout, implement responsive design to adapt to various screen sizes and devices within the HTML document, define typography for readability within the HTML content, apply colors and backgrounds to HTML elements for visual appeal, control spacing with the box model to manage the layout and alignment of HTML elements, and add transitions and animations for user engagement within the HTML interface.  **Responsive Design:**     - Fluid Layouts: Design the interface elements to adapt fluidly to different screen sizes and orientations, ensuring optimal viewing experience on both desktop and mobile devices.     - Media Queries: Utilize CSS media queries to apply different styles based on the device's characteristics, allowing for customized layouts for various screen resolutions.     - Flexible Components: Use flexible grids and components that adjust dynamically to available space, facilitating a consistent and intuitive user interface across devices.     - Viewport Meta Tag: Include the viewport meta tag in your HTML to ensure proper scaling and rendering on mobile devices, enhancing readability and usability.  **Cross-Browser Compatibility:**     - Browser Testing: Test the web interface on multiple browsers such as Chrome, Firefox, Safari, and Edge to ensure consistent rendering and functionality across different platforms.     - Vendor Prefixes: Use vendor prefixes for CSS properties to ensure compatibility with older browser versions that may not support the latest standards.     - Polyfills: Implement JavaScript polyfills for missing functionalities in older browsers, ensuring that critical features work as intended across a wide range of browser environments.     - Feature Detection: Employ JavaScript feature detection techniques to detect browser capabilities and provide fallback solutions or alternative functionalities when necessary.         6. Developing the Mobile Application:  React Native is the framework that the group used for mobile app development. It is cross-platform, meaning that it enables the users to easily address both Android and iOS devices without the necessity of controlling distinct codebases, thereby simplifying the procedure of developing and saving time that is essential.  The group has implemented core functionalities within the JudgeMentor application, ensuring a seamless user experience. With robust user authentication in place, users can securely access the platform and interact with its features. Moreover, the application enables users to update and delete questions and answers, providing them with control over their content. Leveraging geolocation services, the app enhances user engagement by offering location-based features tailored to individual preferences. Real-time communication between the web and mobile platforms further enhances the collaborative experience, enabling instant sharing of insights and feedback.         7. Integrating Mobile and Web Components:  The group has prioritized establishing seamless communication and data synchronization between the mobile app and web interface to ensure a cohesive user experience. By implementing robust backend APIs and utilizing efficient data transfer protocols, the group has enabled real-time data exchange between the two platforms. This ensures that any changes made on either the mobile or web version of the application are promptly reflected across all devices, maintaining data consistency and reducing user friction. Additionally, features have been developed that allow users to seamlessly switch between the mobile and web versions of the application, ensuring continuity in their experience regardless of the device they're using. By maintaining consistent user authentication and session management across platforms, the group has ensured that users can effortlessly transition between devices without interruption. This approach not only enhances user convenience but also reinforces our commitment to delivering a unified and intuitive user experience across all platforms.         8. Creating UML Diagrams:  Introducing Unified Modeling Language (UML) diagrams into our application development process helps to document the system's structure and behavior effectively. By utilizing Class Diagrams, Sequence Diagrams, and Activity Diagrams, the group visualize the application architecture and interactions between components. These diagrams provide a standardized and clear representation, enhancing collaboration and ensuring a well-documented system.  https://lh7-us.googleusercontent.com/hA3VcAtk8S4i6BylUCojzki18iH5MZUioJ10nB-HJkIdGeVl3ue9ultng8WZSsxRRrNSmT4m9ekV4gRi43u7yBq7kDq5XA9HCCArJ-LjJ6B8x_zrUh9JVHl5MSXmO9P05n-XMu0I3NOIQToE9qs5W6s    **Figure 1.8 Sequence Diagram (Web)**  This sequence diagram illustrates the user registration flow and user actions in the system. The user signs up to create an account, then logs in using the created account. In the home screen, the user can add questions and answers as a reviewer, edit and delete questions and answers. Finally, the user logs out of the account.  https://lh7-us.googleusercontent.com/yi-4zvp26ParJZczzRHJ_bxnTw7YzgncPn93wEHpD2Q5YSGiZtQvK5RT_XVFKF1gbKQQiViXmjTni1Po1n85EwQRV03E6B-WwaE5DvGP3rGmD6iE5Os9uOa57Kii_IR2hQEH0gN3xMf1Amj2ETuerQ8  **Figure 1.9 Sequence Diagram (Mobile)**  This sequence diagram illustrates the flow and user actions in the mobile app. The user logs in using the created account. In the home screen, the user can access and view the reviewer which was created on the web. The user can then review the questions and answers and use it as a reviewer. Lastly, the user logs out from the account.    **Figure 2.0 Activity Diagram (Website)**  This activity diagram illustrates the flow and user actions in the website. Firstly, the group has the start and then next , where the user opens the app's website, "JudgeMentor." After that the system will detect or identifies user authentication if the user is not yet register to our system user must sign up, if user appears to be already been registered user can Log in directly, after verifying the user that already been sign up and also the user have been log in already will navigate or return directly to the homepage. In the homepage, users can input questions and answers and if no, users can delete data but if yes, users can successfully add data. When a user appears to edit some data if no user can delete the data that has been inputted, but if yes, the user can successfully update or change some data. Lastly, if a user wants to log out, they will go to the Log in/Sign up page, and if no user can, they can still stay on the home page and then end.   **Figure 2.1 Activity -Diagram (Mobile)**  The activity diagram depicts the flow upon setting up the different environments in the mobile app. The diagram outlines a software development process. It includes sections on group introduction, development environment setup, software development methodology, and mobile/web requirements, design, implementation, testing, deployment, and maintenance. The diagram features elements such as efficient cross-platform tools, personalized learning solutions, and cost-effective development.  https://lh7-us.googleusercontent.com/LFrEsPUYM3K8T0XA5sVcjAmb043YJbYz6bd9ED6lEEjkfKZCcU3wTbiP_Du7Z9zz73isY3zCMMk31Z41Ip_VDX160jl8ql8EJVxSwTt5IPmufJmaVssGS0Muw58-wsGUpkCOFVf5ucTTX5f3K6ZyWKM  **Figure 2.1 Class Diagram (Web and Mobile)**  The content is a text diagram outlining two sections: "JudgeMentor: Your Path to Success" and "Reviewer LearningPaths." The JudgeMentor section includes details about a title, description, benefits, importance, and advantages. The Reviewer LearningPaths section includes information about setup for NodeJS, WaterfallModel, platforms, materials, paths, features, cost-effectiveness, approach, and various setup functions.   1. Integrating Mobile, Web, and Emerging Technologies:   **Database Synchronization**  With the help of database called MySQL that supports synchronization between the web and mobile platforms. This ensures that both the web system and mobile app connect to the same database to maintain data consistency.   1. Testing, Debugging, and Optimization:   The mobile app can be accessed on both android and ios phone devices. The web system can be accessed through different browsers like chrome and Microsoft edge.  **Debugging and optimizing performance issues**    **Figure 2.2 Install APK**  Install the build artifact of the JudgeMentor application to use the app. After the install, you can then use the application.    **Figure 2.3 Throw Application**  Long press the left button of the navigation of your phone and throw or close the application. Then, open back the application.   1. Deployment and Distribution:   **Web Hosting.** The group is yet to choose whether to use major web hosting services like AWS, Google Cloud Platform, or Azure for the reliable deployment of the JudgeMentor web application. Each platform offers robust infrastructure, scalability, and security features. AWS provides extensive services and has strong community support, Google Cloud Platform excels in data analytics and machine learning integrations, and Azure offers seamless integration with Microsoft products. The final decision will depend on the specific requirements and existing infrastructure.  **Server Setup**. Configuring the server environment is crucial. The group needs to ensure that all necessary dependencies and frameworks, such as Node.js, Express, and React, are installed. This involves setting up the server, installing the required software, and configuring the environment to support our application. Using a tool like Docker can help create a consistent environment across different stages of development and deployment, reducing the chances of environment-specific bugs.  **Mobile App Deployment**  Deploying the JudgeMentor mobile app involves distinct steps for both the iOS App Store and Google Play Store. iOS App Store. First, the group must register for the Apple Developer Program to access essential tools for development and distribution. Next, the group should create an app record in App Store Connect, providing metadata, screenshots, and descriptions. In Xcode, the group must configure the app to meet Apple’s guidelines, setting up provisioning profiles and entitlements. Once configured, the group may submit the app for review, addressing any feedback from Apple to ensure compliance and quality.Google Play Store. For the Google Play Store, the group starts by registering for a Google Developer Account. Using the Google Play Console, the group creates an app listing, including necessary details and visuals. The group then prepare the app bundle or APK, ensuring it meets Google’s requirements for API levels and content policies. Finally, the group will submit the app for review, ready to make any necessary adjustments based on Google’s feedback.Marketing and Community EngagementBlog Posts and Articles. Writing and publishing blog posts and articles about the features and benefits of JudgeMentor can significantly increase the group's reach. Targeting educational and tech blogs can help connect with the primary audience. These articles should highlight unique features, use cases, and success stories to attract potential users and partners.Social Media Campaigns. Utilizing platforms like Facebook, Twitter, Instagram, and LinkedIn for social media campaigns can help create buzz around JudgeMentor. Sharing content, updates, and success stories can engage the audience and foster a community around the application. Regular posts, interactive content, and targeted ads can enhance the group's online presence.Forums and Groups. Participating in relevant forums and online communities, such as Reddit, Quora, and specialized education groups, is a great way for the group to share insights and gather feedback. Engaging with these communities can provide valuable input on user needs and pain points, allowing for continuous improvement of JudgeMentor. Additionally, this helps in building a supportive user base and establishing authority in the educational tech space.  1. Presentation and Documentation:   This step will take place in the actual presentation of the project.   1. Reflection and Discussion:   Creating this project was challenging for the group as it applied lessons from previous subjects. The group encountered difficulties, but with the collaborative efforts of its members, the project was successfully completed.  **Group Contribution**  **Web and App Development**  Erussha Mae Bangcoyo - Front-end Developer  Alisha Nicole Salarda - Front-end Developer  Michelle Jean Arot - Back-end Developer  Kate Royce Ternate - Back-end Developer  **Documentation**  Erussha Mae Bangcoyo - Created the class diagram and sequence diagram  Michelle Jean Arot - Created the DFD context diagram level 0, context diagram level 1, and child diagram  Alisha Nicole Salarda - Created the use case diagram and sequence diagram  Kate Royce Ternate - Created the activity diagram, and class diagram  The rest of the parts and steps of the documentation that were not mentioned were done as a group which means that the group completed the steps together and with the help of each member.  In software development, Data Flow Diagrams (DFDs) play a crucial role in requirement analysis and system design by providing a visual representation of how data flows through a system. They help stakeholders, including developers and non-technical participants, to understand and communicate the processes, data stores, and data movements within the application. DFDs serve as a blueprint during the design phase, guiding the structuring of data handling and processing components and facilitating the identification of potential issues early in the development process. By clarifying how data is processed and transferred, DFDs ensure that both the system requirements and design are well understood and documented.  Unified Modeling Language (UML) diagrams are another essential tool in software development, offering a standardized way to visualize the architecture, design, and implementation of a system. UML diagrams, such as use case diagrams, class diagrams, and sequence diagrams, help developers and stakeholders to conceptualize and communicate the system's structure and behavior. They provide a detailed and organized method to capture functional and non-functional requirements, model different aspects of the system, and ensure that all components and their interactions are clearly defined. UML diagrams support better planning and coordination throughout the development lifecycle, facilitating design consistency, code quality, and maintainability.  Emerging technologies significantly impact software development by introducing new tools, frameworks, and methodologies that enhance efficiency, performance, and innovation. Technologies like artificial intelligence (AI), machine learning (ML), blockchain, and the Internet of Things (IoT) enable developers to create more intelligent, secure, and connected applications. AI and ML algorithms can be used to automate tasks, predict user behavior, and enhance decision-making processes. Blockchain provides robust security and transparency for transactions and data management, while IoT allows for seamless integration of physical devices with software applications. By leveraging these emerging technologies, developers can build more advanced, scalable, and user-centric software solutions that meet the evolving needs of the market. | | | |
| 6ffActivity No : | # 2 | | |
| Activity Name : | Comprehensive Cross-Platform Application Development Laboratory: Exploring Mobile, Web, and Emerging Technologies with DFD and UML Documentation | | |
| Student ID : | 2021303252, 2021303243, 2021303252 , 2021300814 | | |
| Student Name : | Erussha Mae Bangcoyo , Michelle Jean Arot , Alisha Nicole Salarda , Kate Royce Ternate | | |
| Course & Year | BSIT – 3rd Year | | Section : 3R4 |