ETHIOPIAN INTERNATIONAL JOURNAL OF MULTIDISCIPLINARY RESEARCH

eISSN: 2349-5715 pISSN: 2349-5707

Volume: 11, Issue 06, June-2024 SJIF 2019: 4.702 2020: 4.737 2021: 5.071 2022: 4.919 2023: 6.980 2024: 7,662

DIGITAL TECHNOLOGIES IN TEACHING GENERAL PROFESSIONAL DISCIPLINES

Nurılla Orınbetov
Associate Professor,
Kamilova Marzhangul Karamdinovna
senior teacher.
Gulchekhra Joldasova
1st year doctoral student of the
Department of Technological Education.
Nukus State Pedagogical Institute named after Ajiniyaz.

Annotation: This article examines the analysis of methods and algorithms that can be used for the most effective integration of web applications into the educational process. This is an analysis of the advantages and disadvantages of using Flask to create mobile applications. Flask is comparable to other web frameworks such as Django and Fast API in terms of ease of development, performance, extensibility, and community support.
Keywords: web framework, web applications, digital technologies, Flask Django and Fast API web frameworks, Flask microframework, encryption and session management, use of HTML, JavaScript.

The use of web applications in the educational process provides unique opportunities for creating interdisciplinary connections, facilitates access to educational resources, and fosters the development of critical thinking and independent learning among students. Within the scope of this study, special attention is given to analyzing the methods and algorithms that can be used for the most effective integration of web applications into the teaching process with the aim of ensuring deep interaction between general professional and specialized disciplines.
In the modern world, where digital technologies permeate every corner of our lives, mobile applications become an integral part of everyday life. An increasing number of companies and individual developers strive to create functional, convenient, and responsive mobile applications to meet the constantly growing needs of users. In this context, Python stands out as one of the preferred programming languages for developing web applications due to its flexibility and powerful set of tools. The Flask web framework, written in Python, offers a lightweight yet powerful platform for creating web applications, including mobile applications. Flask is designed for developing mobile applications using everything from tool selection and development environment to implementing basic functional requirements and testing applications.
Flask is a micro-framework for web development that allows developers to focus on what really matters for their applications while simultaneously reducing the amount of code needed to get started. This feature makes it an ideal choice for mobile application development, where speed of development and ease of maintenance are key factors. Within the scope of this work, the process of developing a mobile application using Flask, including setting up the development environment, designing the application architecture, implementing the user interface and server logic, as well as application performance, will be examined in detail.
This is an analysis of the advantages and disadvantages of using Flask for creating mobile applications. Flask is compared with other web frameworks such as Django and FastAPI in terms of ease of development, performance, scalability, and community support. Furthermore, special attention is given to the integration of the mobile application with modern technologies such as databases, caching systems, and cloud platforms.

eISSN: 2349-5715 pISSN: 2349-5707
Volume: 11, Issue 06, June-2024 SJIF 2019: 4.702 2020: 4.737 2021: 5.071 2022: 4.919 2023: 6.980 2024: 7,662

Let's continue our discussion on the development, choice of tools, and setting up the development environment for a mobile application to integrate general professional and specialized disciplines in Flask.

First and foremost, choosing the right set of tools and technologies is a crucial step in the development of any software product. This choice for developing a mobile application with Flask includes not just the web framework itself, but also additional libraries and services that provide application functionality and enhance performance. At this stage, developers must select a database, version control system, testing and debugging tools, as well as a platform for deploying applications.

Setting up the development environment starts with installing Python and Flask. To separate the project's dependencies from the global Python environment, it's recommended to use a virtual environment (virtualenv), which prevents conflicts between libraries and simplifies dependency management. Installing Flask and necessary libraries is done using the package manager pip. For database work, Flask extensions can be used, such as Flask-SQLAlchemy for integration with SQLAlchemy, offering a convenient interface for working with relational databases, or Flask-MongoEngine for working with NoSQL databases like MongoDB. After choosing the technology stack, the next step is to develop the application architecture. The architectural approach to development largely determines the flexibility, scalability, and ease of maintenance of the final product. Flask is a microframework that does not impose strict architectural restrictions, giving developers the freedom to choose their architectural style. However, this means that the responsibility for creating a precise and efficient architecture falls on the developers. A modular approach, which involves dividing the application into independent modules, each responsible for a specific part of the functionality, simplifies development and testing, as well as facilitates future expansion of the application.

Security concerns occupy a central place in the development of mobile applications. Flask provides several mechanisms to ensure application security, including protection against Cross-Site Request Forgery (CSRF) and Cross-Site Scripting (XSS), data encryption, and session management. It's crucial to carefully consider the user authentication and authorization system to protect personal data and prevent unauthorized access to application features.

User Interface (UI) development plays a key role in creating mobile applications, as the user interacts with the app. When developing the UI in the context of Flask, it's important to focus on creating an adaptive and intuitive design that adjusts to various screen sizes of mobile devices. Using HTML, CSS, and JavaScript is the standard approach to developing web interfaces and simplifies the integration of these technologies with server-side logic through Flask's Jinja2 template system. Jinja2 templates allow developers to efficiently create dynamic HTML content using information obtained from the server, enabling the creation of interactive user interfaces without page reloads.

Ensuring the performance of mobile applications is also a crucial part of development. Various optimization methods can be used for this purpose, such as minimization and compression of resources (CSS and JavaScript files), asynchronous resource loading, and caching. Flask supports the use of extensions like Flask-Assets, which automate the optimization process of static files, thereby improving the loading speed of applications and the overall user experience.

Moving to server-side logic, Flask offers developers a flexible routing system for handling client requests. Defining routes and their corresponding handler functions allows developers to easily manage the flow of data between the client and server, ensuring logic for controlling various user actions in the application. Integration with databases and other external services is facilitated through Flask modules and extensions, simplifying the creation of complex web applications with rich functionality.

eISSN: 2349-5715 pISSN: 2349-5707
Volume: 11, Issue 06, June-2024 SJIF 2019: 4.702 2020: 4.737 2021: 5.071 2022: 4.919 2023: 6.980 2024: 7,662

Application security remains a crucial aspect in the implementation of server-side logic. Flask provides built-in mechanisms to protect against common threats such as SQL injections and CSRF attacks, and Flask-Security leverages additional extensions for authentication, authorization, and user session management.

When developing a mobile application with Flask, testing is as important as developing functionality. Unit testing, integration testing, and user interface testing allow developers to identify and fix errors at an early stage, ensuring the high quality of the final product. Flask supports integration with popular testing frameworks like pytest and Flask-Testing, simplifying the process of creating and running tests.

On this page, we've covered key aspects of developing the user interface and server-side logic of a mobile application in Flask, including the importance of performance optimization, security measures, and testing.

Ensuring the stability and reliability of the application before its launch is crucial. The use of automated testing tools, such as pytest or Flask-Testing, can greatly simplify this process, allowing developers to efficiently test the performance of individual application components as well as their interactions.

During the testing phase, special attention is given to verifying the functionality of all user interface elements as well as the functionality implemented on the server side. Unit tests allow for isolating and testing the performance of individual functions and methods of the application, while integration tests assess the interaction between different modules and levels of the application, ensuring their correct integration. User interface testing, often performed using web interface testing automation tools such as Selenium, helps ensure that the application displays and operates correctly across different browsers and devices.

Beyond functional testing, performance and security tests play a crucial role. Performance testing can help identify bottlenecks in your application that may slow it down or increase server load. Using tools like LoadRunner or JMeter allows for simulating simultaneous access by a large number of users, thus testing the application's ability to maintain high performance and stability under heavy load. Security testing aims to identify potential vulnerabilities in applications that attackers could exploit to gain unauthorized access to data or application functions. Tools like OWASP ZAP or Burp Suite enable developers to comprehensively test for application vulnerabilities and security weaknesses.

The final stage of mobile application development is its deployment and maintenance. Deployment involves choosing a suitable hosting platform, setting up server infrastructure, and publishing the application. Depending on the application's performance and scalability requirements, developers can choose between traditional web hosting, cloud services, or Platform as a Service (PaaS) options like Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, Heroku, or DigitalOcean App Platform. Each of these options offers its advantages and disadvantages, including differences in cost, ease of management, and scalability.

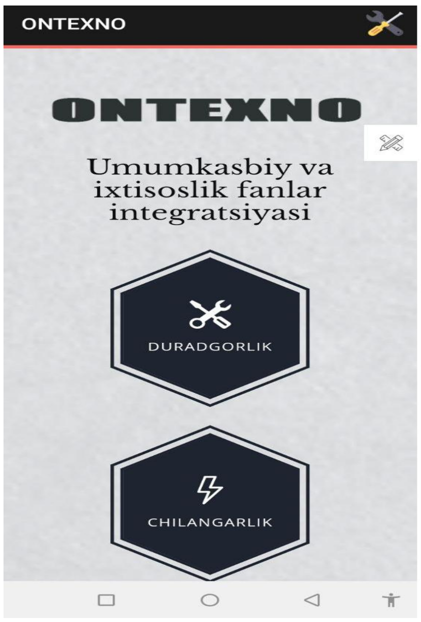
After the application launch, the support and update phase begins. It's important to regularly update the application, fixing detected bugs, improving its functionality and security. Receiving user feedback is a key aspect of this process, as it allows identifying the application's shortcomings and determining further development directions. Using project management and issue tracking systems, such as Jira or GitHub Issues, simplifies project work organization, enabling effective task planning, tracking their completion, and interacting with the development team.

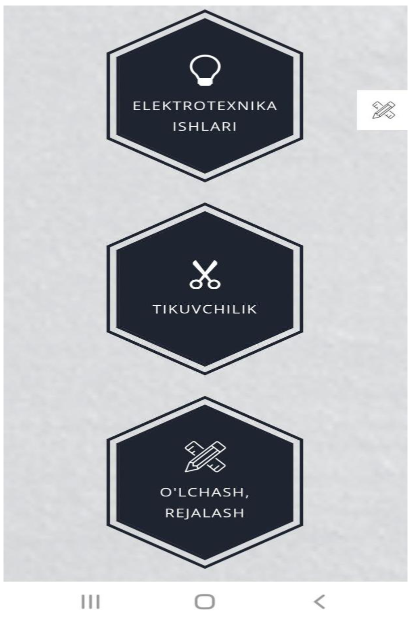
On this page, we have covered the fundamental aspects of testing, deployment, and maintenance of a Flask mobile application. These stages are an integral part of the application development lifecycle, ensuring its quality, reliability, and alignment with user expectations.

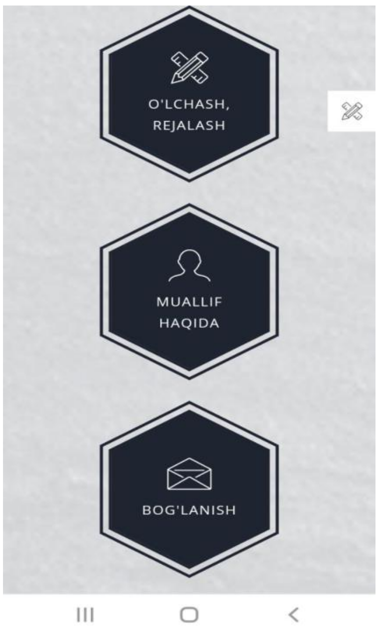
eISSN: 2349-5715 pISSN: 2349-5707
Volume: 11, Issue 06, June-2024 SJIF 2019: 4.702 2020: 4.737 2021: 5.071 2022: 4.919 2023: 6.980 2024: 7,662

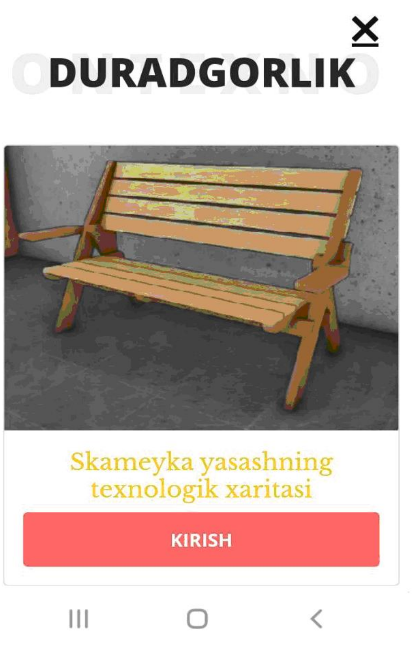
Let's delve deeper into application optimization and promotion strategies.
Moving to the next crucial aspect of Flask mobile application development, we will focus on strategies for optimizing the application and promoting it in the market.
Application optimization is a key factor affecting its performance, speed, and overall user interaction. In the context of a Flask-developed mobile application, optimization can include a range of actions aimed at improving page load times, enhancing server efficiency, and minimizing delays in user interaction with the application.
One optimization method is the use of caching on both the server and client sides. Caching can reduce the number of requests to the server for frequently changing data, thereby reducing server load and speeding up page loading for the user. Flask offers several caching extensions, such as Flask-Caching, which can be easily integrated into your application and provide flexible settings for cache management.
Another aspect of optimization is minimizing and bundling static files (CSS and JavaScript), which reduces the number of HTTP requests needed to load a page and decreases the total volume of data transferred. Tools like Webpack or Gulp can be used alongside Flask to automate these processes.
Adaptive design of the web page also plays a crucial role in optimization, allowing the application to display and function correctly on various devices and screen sizes. Utilizing a CSS framework, such as Bootstrap or Foundation, can significantly simplify adaptive design.
After optimizing your application, the next step is its promotion. Regardless of the application's quality, without an effective marketing strategy, it might get lost among the multitude of other apps. Promotion can include the use of social media, contextual advertising, SEO optimization to increase the application's visibility in search engines, and engaging with user reviews on various platforms and forums.
One crucial aspect of application optimization is the efficient use of the database. This includes proper database schema design, using indexes to speed up data searches, and optimizing queries to minimize query execution time. Developers should regularly analyze database performance using profiling tools to identify and address issues. Frequently used data caching can significantly enhance application speed by reducing the number of database calls.
Page load time is an important factor for user interaction, especially in mobile web applications. To reduce it, you can employ methods such as compression and bundling of static files (CSS and JavaScript), asynchronous loading of resources, and image optimization. Using modern image formats like WebP can significantly reduce image size and ensure faster page loading while maintaining quality. Additionally, implementing Lazy Loading technology for images and videos allows deferring the loading of these elements until they

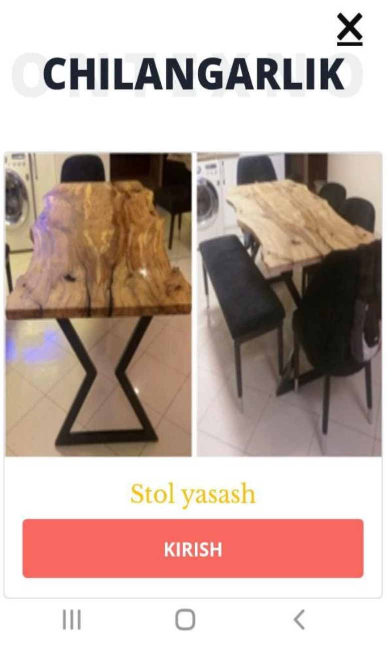
eISSN: 2349-5715 pISSN: 2349-5707
Volume: 11, Issue 06, June-2024 SJIF 2019: 4.702 2020: 4.737 2021: 5.071 2022: 4.919 2023: 6.980 2024: 7,662

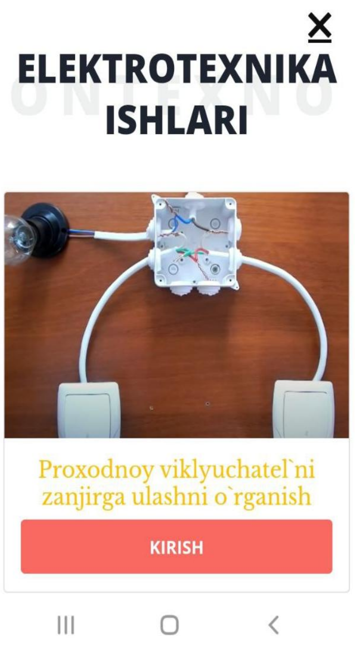




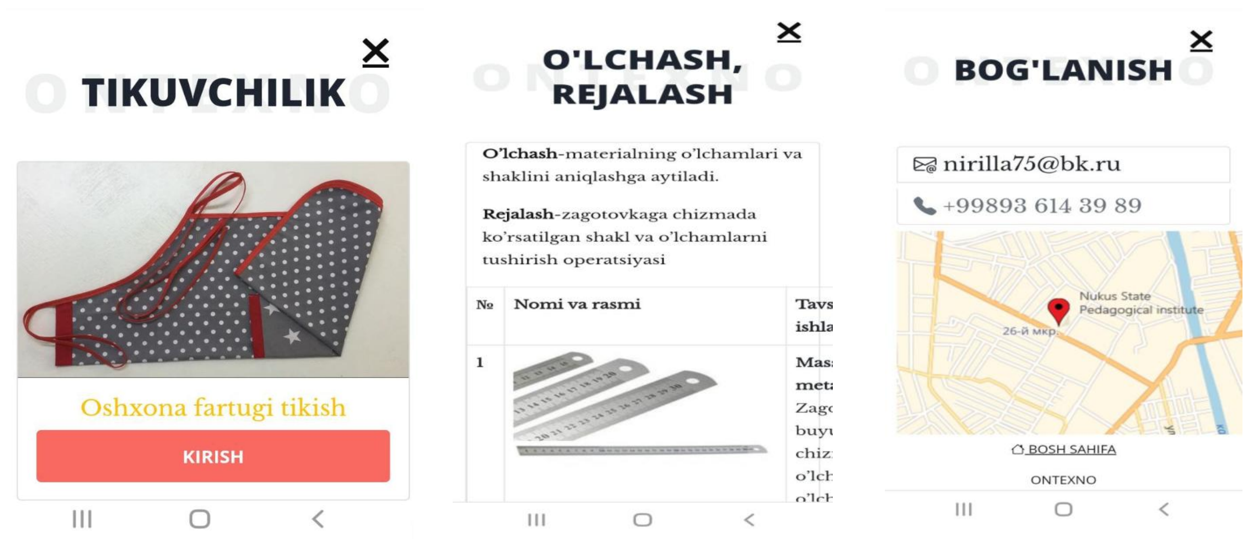








eISSN: 2349-5715 pISSN: 2349-5707
Volume: 11, Issue 06, June-2024 SJIF 2019: 4.702 2020: 4.737 2021: 5.071 2022: 4.919 2023: 6.980 2024: 7,662



O'lchash-materialning o'lchamlari va shaklini aniqlashga aytiladi.

Rejalash-zagotovkaga chizmada ko'rsatilgan shakl va o'lchamlarni tushirish operatsiyasi

№	Nomi va rasmi	Tavs ishla
1		Mas met Zag buyı chiz o'lch o'lch

become visible to the user, thereby accelerating the initial page load.

To enhance performance on the client side, developers utilize frameworks like React or Vue, which provide efficient DOM updates and an improved user interface through a component-based approach. These modern JavaScript frameworks and libraries facilitate the creation of dynamic, responsive user interfaces that can significantly improve the user experience. It's also important to pay attention to the performance of JavaScript code to avoid costly operations and memory leaks.

Adaptive design is essential for correctly displaying the application across different devices and ensuring ease of use. The use of responsive layouts, media queries, and adaptive images helps achieve this goal by adjusting the application interface to the screen resolution and orientation of the device. This is particularly crucial for mobile applications, where users may employ a wide range of devices with various screen sizes.

Regular performance tests are an integral part of the optimization process. Using tools such as Google PageSpeed Insights, Lighthouse, and WebPageTest can help assess page load speed, resource efficiency, and application accessibility. These tools provide specific performance recommendations that developers can use to optimize their applications.

The conclusion showed that integrating web applications into the educational process is a powerful tool for improving the quality of education, increasing motivation and engagement among students, developing critical thinking, and problem-solving skills. Web applications help create a more flexible and inclusive learning environment, open to students with diverse needs and preferences.

REFERENCE

1. Tulaganov Z. Sh. “Masofaviy ta’limda qo‘llaniladigan mobil qurilmalar” Academic Research in Educational Sciences Volume 4 | Issue 6 | June 2023 yil https://cyberleninka.ru/article/n/masofaviy-ta-limda-qo-llaniladigan-mobil-qurilmalar
2. Qingkai Kong, Timmy Siauw, Alexandre M.Bayen – “Python Programming and Numerical Methods A Guide for Engineers and Scientists”; 2021 Elsevier Inc;