Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Experiment 10

Aim: To study and implement deployment of ToDo PWA to GitHub Pages.

Theory:

GitHub Pages as a PWA Hosting Platform

GitHub Pages functions as a static content delivery network (CDN), with built-in behaviors that directly impact Progressive Web App (PWA) deployment:

- Static File Model: Only delivers pre-built static assets (HTML, CSS, JS) with no support for server-side logic.
- Automatic HTTPS: Provides TLS encryption by default, fulfilling PWA security requirements.
- JAMstack-Oriented: Adheres to the JavaScript, APIs, and Markup model for frontend-backend decoupling.

PWA Deployment Considerations

To operate seamlessly on GitHub Pages, a PWA must align with its platform constraints:

- Client-Side Rendering (CSR): Utilization of CSR patterns via frameworks (e.g., React, Vue) or native JavaScript to handle routing and dynamic rendering.
- API Integration: Requires CORS-compliant external APIs for real-time data fetching.
- Resource Linking: Careful handling of absolute and relative paths to ensure proper asset loading in subdirectory-hosted repositories.

Service Worker Design Constraints

Service worker integration must be carefully scoped and optimized:

- Scope Definition: Registration path must align with the GitHub Pages directory structure.
- Caching Strategy:
  - Use cache-first for static resources to improve offline capability.
  - Apply network-first for frequently updated dynamic content or API responses.
- Versioning & Cache Busting: Leverage file hashing during build to invalidate old cached versions.

Deployment Workflow

Repository Setup:

- Specify the target branch (main or gh-pages)
- Define the build output directory (e.g., /build, /dist)
- Configure custom domain (if applicable)

Build Optimization:

- Pre-compile and minify static assets

- Generate route manifests for routing
- Preload critical resources for performance

Deployment Activation:
- Enable GitHub Pages under repository settings
- Integrate CI/CD pipelines for automated builds and deployments

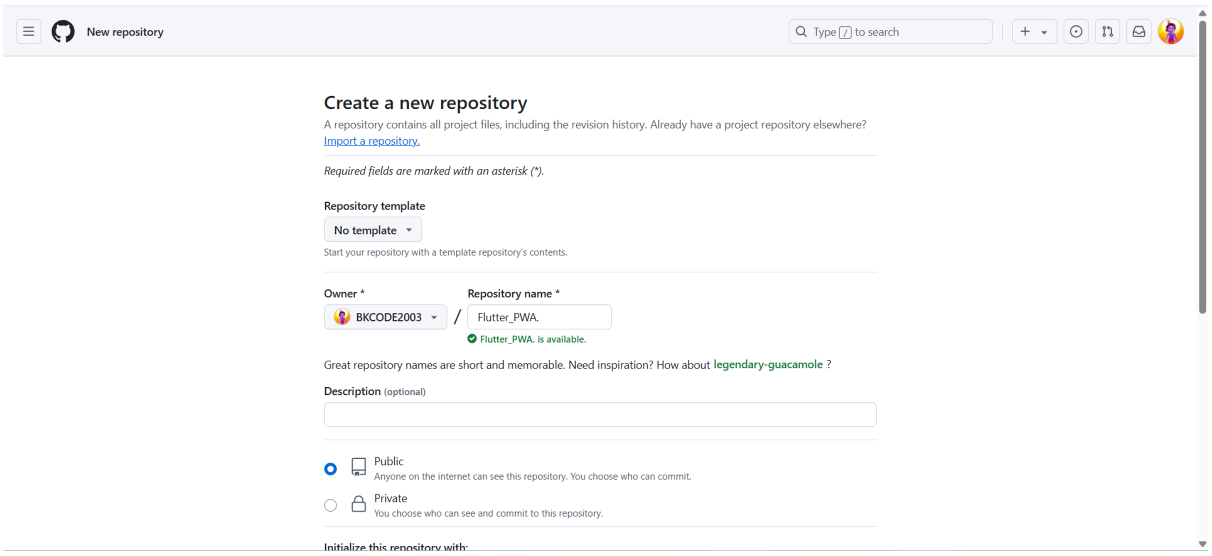
Network Characteristics
- Cache-Control Policies: Governed by GitHub's default headers (e.g., long cache lifetimes unless manually configured)
- CDN Distribution: Assets are distributed globally, with some propagation delay
- HTTP/2 Compatibility: Enables multiplexing for improved loading speed

Testing & Validation
- Lighthouse Reports: Evaluate PWA metrics like performance, accessibility, and offline readiness
- Offline Testing: Simulate slow or no-network conditions to validate cache fallback
- Cross-Browser Checks: Ensure consistent behavior and compatibility across major browsers

Platform Limitations
- No Backend Support: All logic must be handled on the client side
- Subdirectory Hosting Complexity: Resource paths must be adapted when apps are served from nested URLs
- Size Restrictions: GitHub imposes a soft limit of 1GB per repository

Steps:

1. **Create a new github repository for the project.**



Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository.

Required fields are marked with an asterisk (*).

Repository template: No template

Start your repository with a template repository's contents.

Owner *: BKCODE2003 / Repository name *: Flutter_PWA.

Flutter_PWA. is available.

Great repository names are short and memorable. Need inspiration? How about legendary-guacamole ?

Description (optional)

Public — Anyone on the internet can see this repository. You choose who can commit.

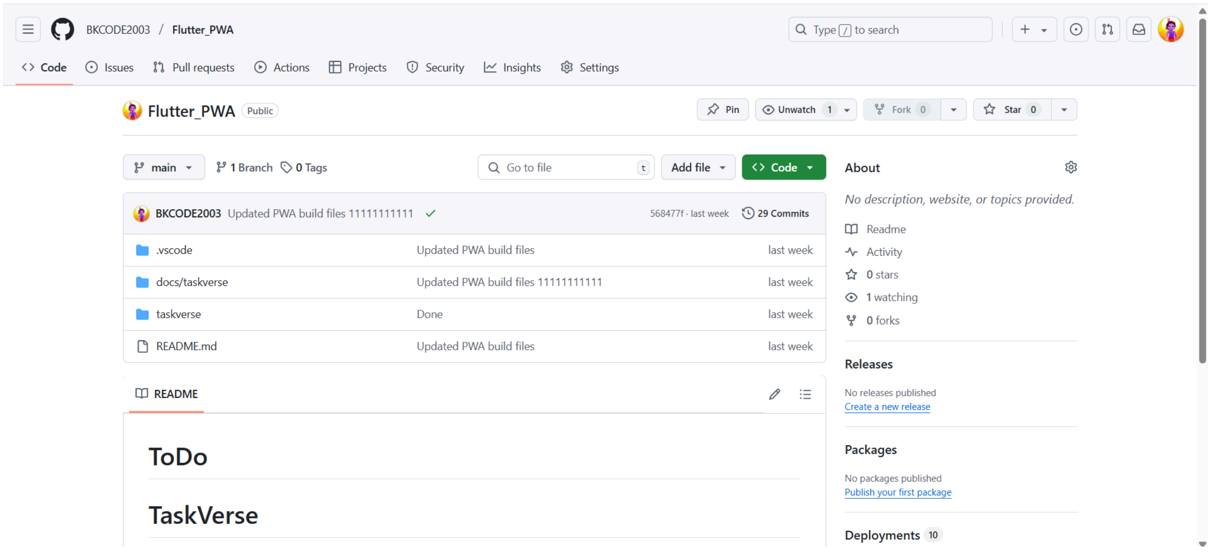
Private — You choose who can see and commit to this repository.

Initialize this repository with:

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

2. Commit all the files to the repository



BKCODE2003 / Flutter_PWA

Flutter_PWA Public

main · 1 Branch · 0 Tags

BKCODE2003 Updated PWA build files 11111111111 ✓ 568477f · last week · 29 Commits

.vscode	Updated PWA build files	last week
docs/taskverse	Updated PWA build files 11111111111	last week
taskverse	Done	last week
README.md	Updated PWA build files	last week

README

ToDo

TaskVerse

About

No description, website, or topics provided.

Readme · Activity · 0 stars · 1 watching · 0 forks

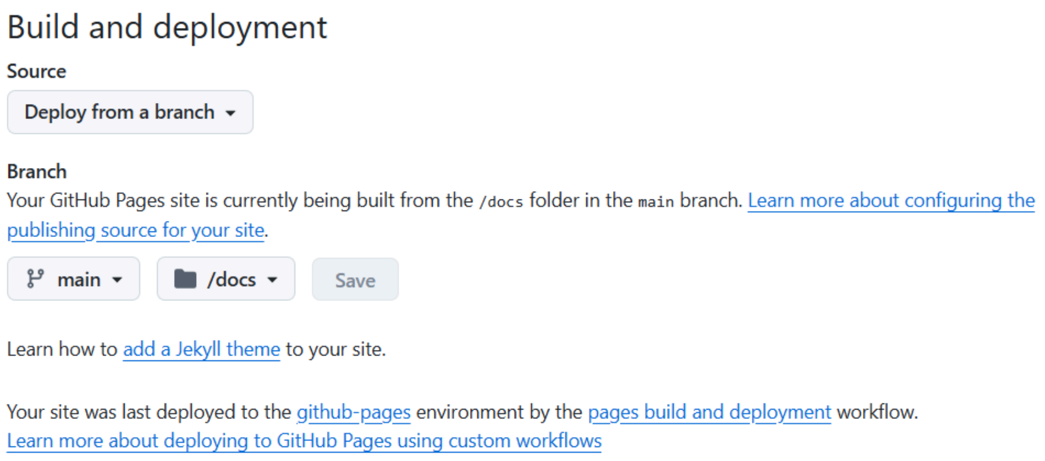
Releases: No releases published. Create a new release

Packages: No packages published. Publish your first package

Deployments 10

3. Navigate to Settings -> Pages

4. Select source as "Deploy from a branch" and branch as the branch where your complete pwa app is.



Build and deployment

Source

Deploy from a branch

Branch

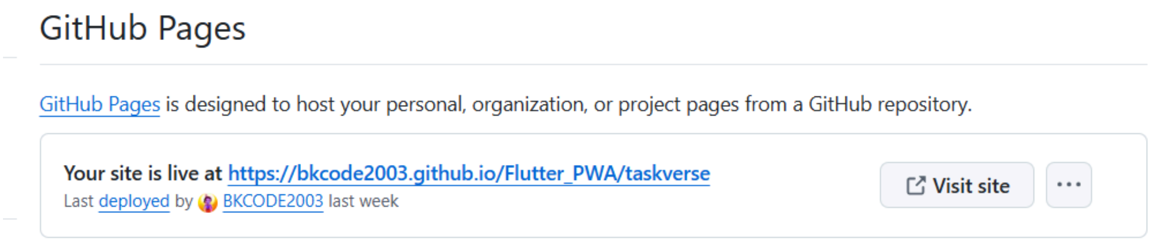
Your GitHub Pages site is currently being built from the `/docs` folder in the `main` branch. Learn more about configuring the publishing source for your site.

main · /docs · Save

Learn how to add a Jekyll theme to your site.

Your site was last deployed to the github-pages environment by the pages build and deployment workflow.
Learn more about deploying to GitHub Pages using custom workflows

5. After saving, wait for some time, and then reload the page. You will see the link for the hosted website.



GitHub Pages

GitHub Pages is designed to host your personal, organization, or project pages from a GitHub repository.

Your site is live at https://bkcode2003.github.io/Flutter_PWA/taskverse

Last deployed by BKCODE2003 last week

PWA APP Link: https://bkcode2003.github.io/Flutter_PWA/taskverse/

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

GitHub Link: https://github.com/BKCODE2003/Flutter_PWA

Conclusion: This experiment successfully showcased the deployment of a Progressive Web App (PWA) on GitHub Pages, proving it to be a viable, zero-cost hosting solution for production-ready PWAs. By carefully configuring build settings, managing asset paths, and optimizing the service worker, we confirmed that GitHub Pages fully supports essential PWA features such as offline capabilities, HTTPS security, and installability. The deployment process emphasized the significance of static site architecture and demonstrated that GitHub's infrastructure meets the technical requirements for hosting high-performance, secure progressive web applications with dependable global CDN distribution.