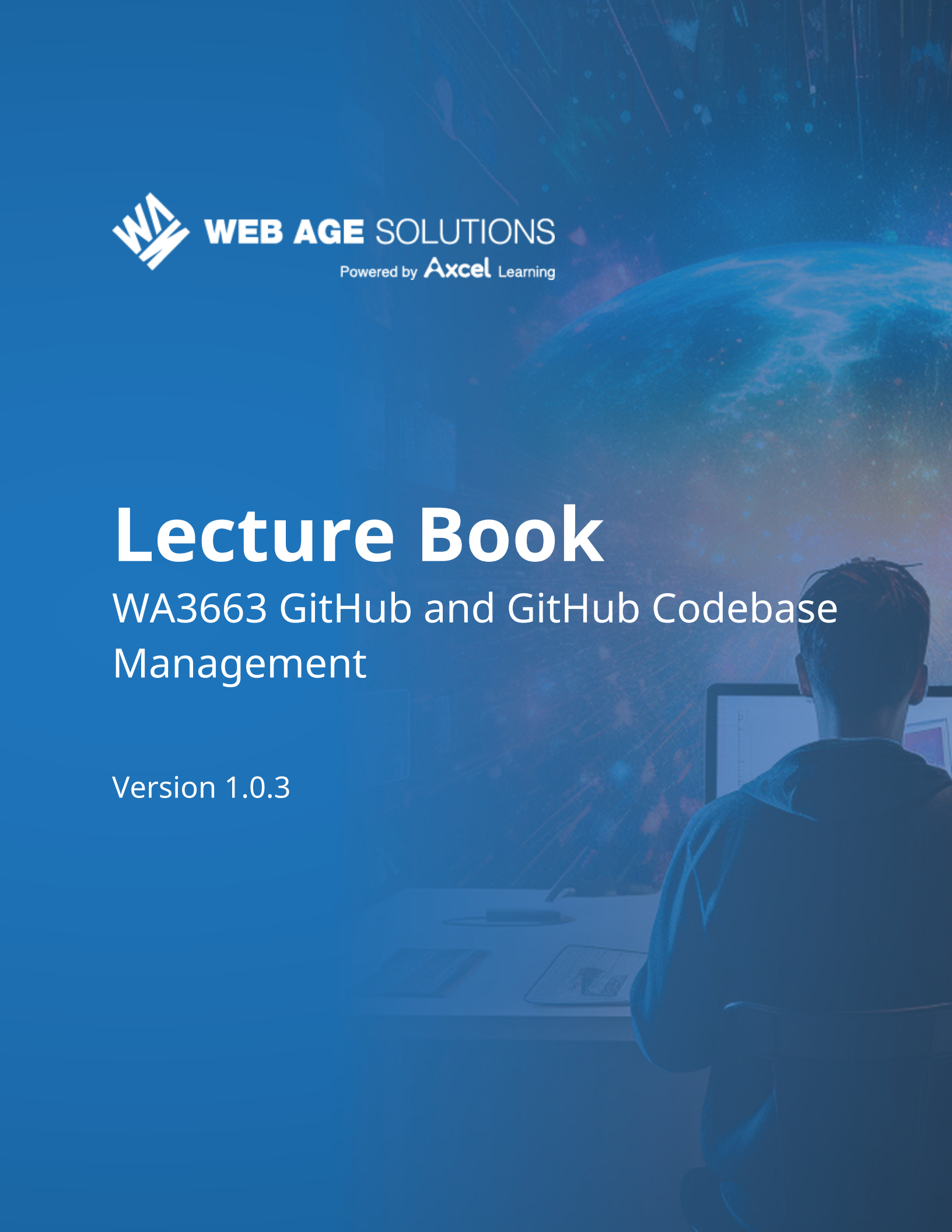


Lecture Book

WA3663 GitHub and GitHub Codebase Management

Version 1.0.3

Revision 1.0.3 published on 2025-01-30.

Web Age Solutions, LLC
1 California Street Suite 2900
San Francisco, CA 94111
https://www.webagesolutions.com

USA: 1-877-517-6540, email: getinfousa@webagesolutions.com
Canada: 1-877-812-8887 toll free, email: getinfo@webagesolutions.com

GitHub and GitHub Codebase Management

Chapter 1 – GitHub

1.1 What is GitHub?

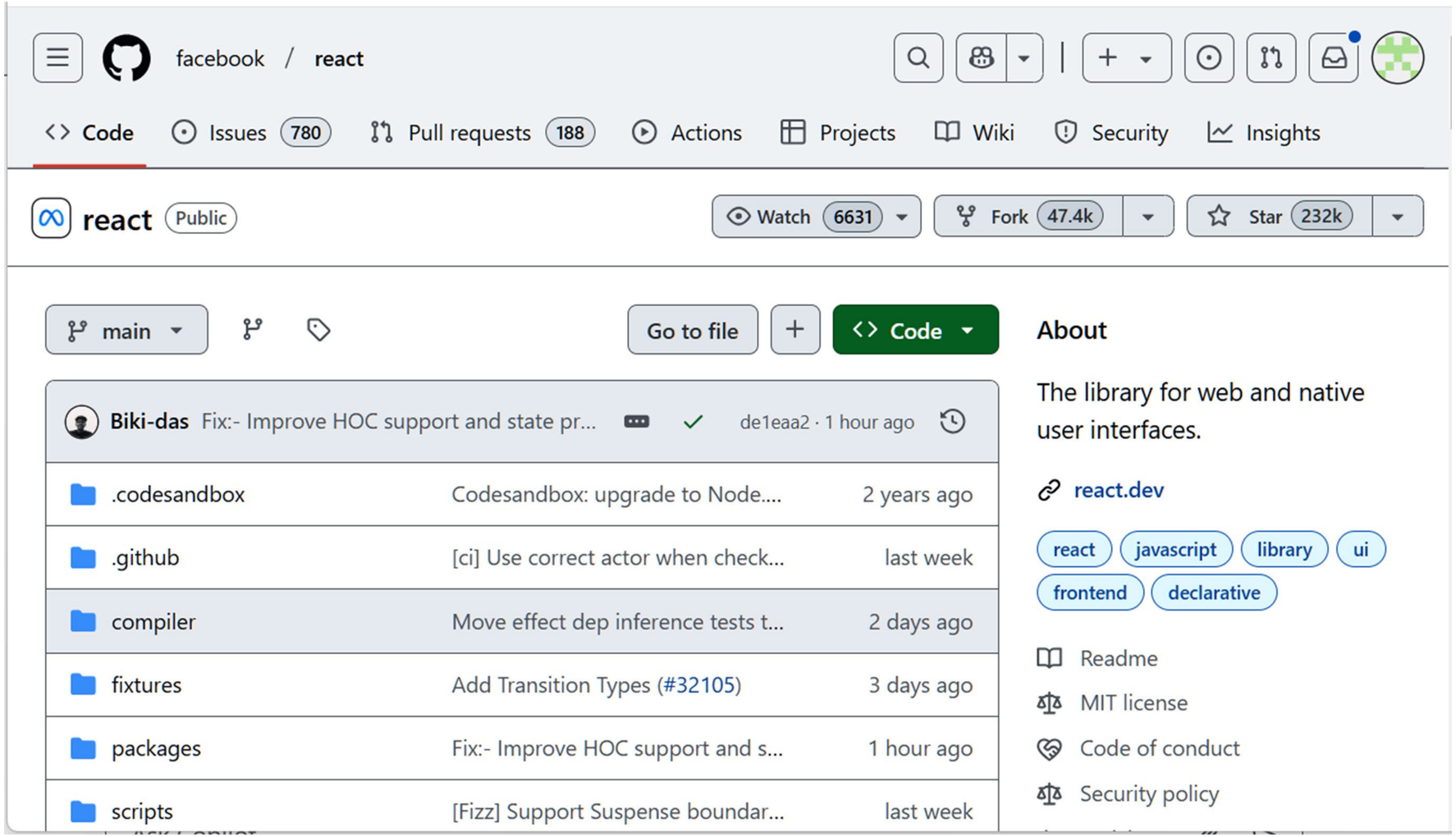
GitHub is:

- A web site for hosting and managing Git repositories.
- A tool for storing, tracking and collaborating on code.
- A shared workspace for developers.
- An environment that supports open-source projects.
- A platform for building, testing and deploying software.

1.2 GitHub - Features

- **Version Control:** Tracking changes to code over time
- **Collaboration:** Enables multiple developers to work on the same codebase
- **Code Reviews:** Facilitates peer feedback and improved code quality
- **Issue Tracking:** Helping manage bugs, feature requests, and other tasks
- **Pull Requests:** Proposing and reviewing code changes before merging them into the codebase
- **Branching:** Creating separate lines of development
- **Cloning:** Creating a local copy of a repository that can be synced with the shared repo
- **Project Management:** Tools for planning, organizing, and tracking a project's progress
- **Integrations:** Connecting with other tools and services in the development workflow

1.3 GitHub - Repository Example



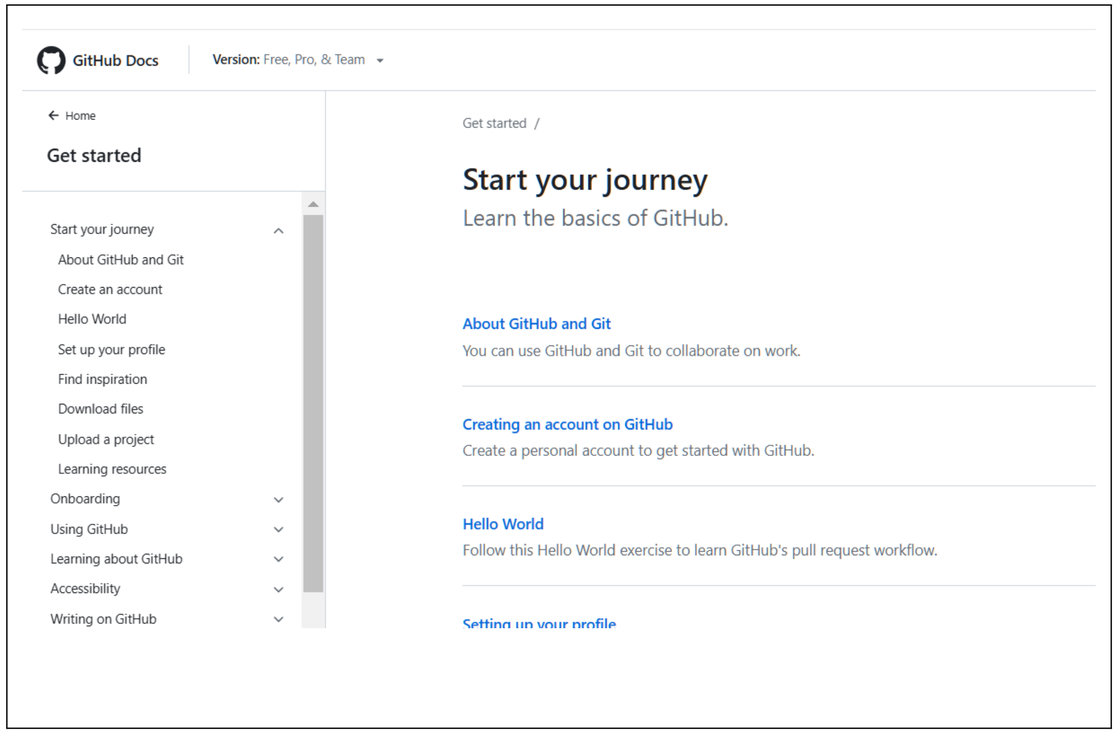
1.4 GitHub –User Interface Tabs

- **Code:** The heart of the GitHub, where you store and manage all your project files
- **Issues:** A to-do list for your project, for tracking bugs, feature requests, and other tasks
- **Pull Requests:** A system to propose changes to the codebase. You create a pull request, other developers review it, and then they can be merged into the main project
- **Actions:** Automate tasks like running tests or deploying your code
- **Projects**: Organize your work using kanban boards, to-do lists, and other tools
- **Wiki:** A knowledge base for storing important project information and documentation
- **Security:** Features to help you identify and fix vulnerabilities in your code
- **Insights:** Visualizations that show how your project is being developed and used

1.5 Start your Journey

In-depth articles covering many GitHub related topics

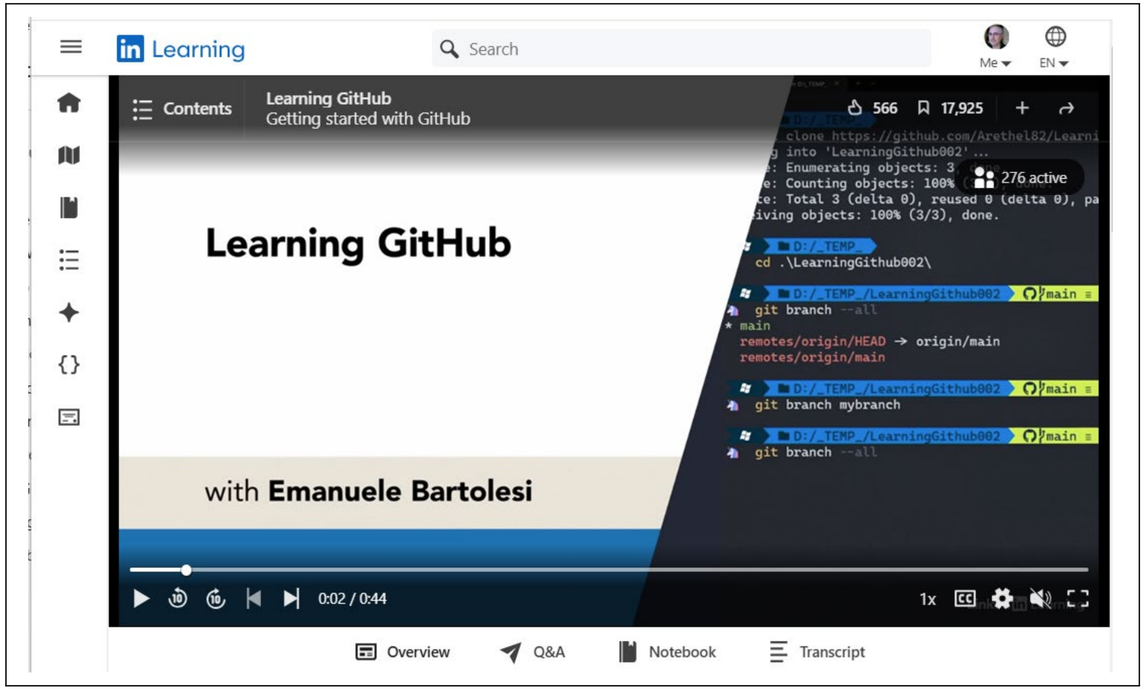
https://docs.github.com/en/get-started/start-your-journey



1.6 Learning GitHub

Video course. Review of GitHub features and functions

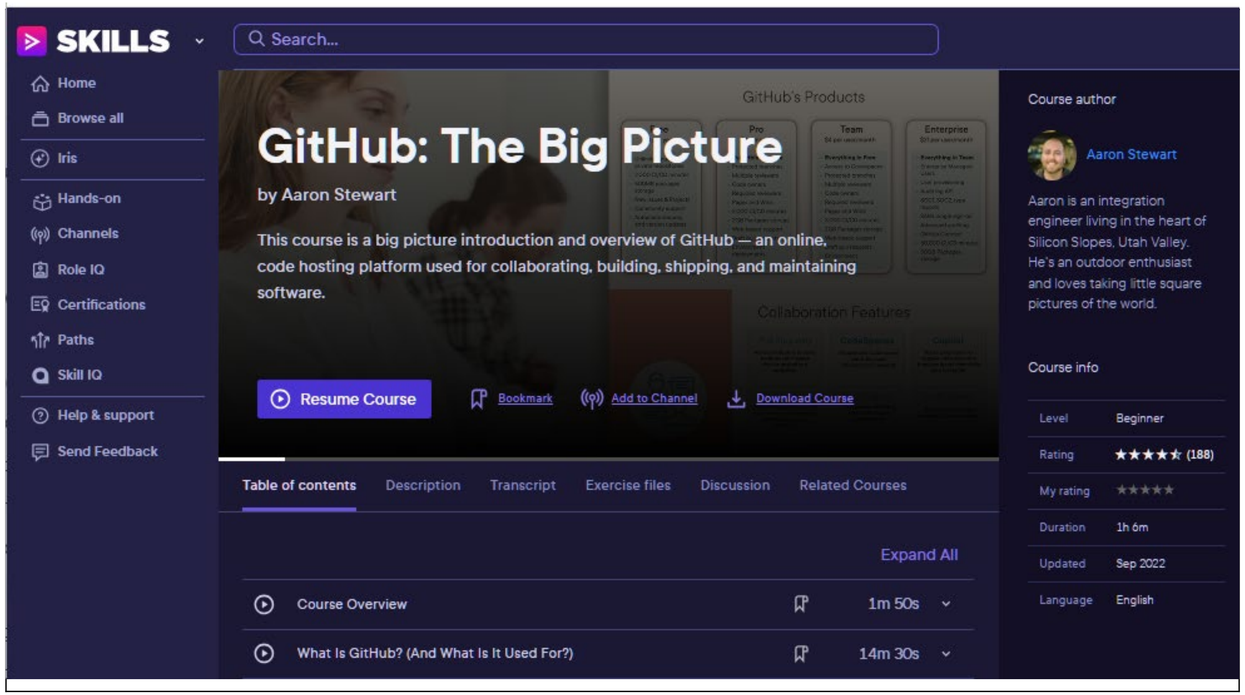
https://www.linkedin.com/learning/learning-github-18719601



1.7 GitHub: The Big Picture

Pluralsight video course. Provides an overview of GitHub.

https://app.pluralsight.com/library/courses/github-big-picture/table-of-contents



GitHub and GitHub Codebase Management

Chapter 2 – GitHub Codebase Management

2.1 Codebase Management – Definition

"Codebase management refers to the practices, processes, and tools used to control, maintain, and organize the source code of a software project. It can work on the codebase simultaneously with involves tracking the changes made to the code, ensuring that different developers out conflicts, and maintaining the quality and integrity of the code over time."

2.2 Key Aspects of Codebase Management – List

1. Version Control
2. Branching and Merging
3. Code Review
4. Dependency Management
5. Continuous Integration and Continuous Deployment (CI/CD)
6. Issue Tracking
7. Documentation
8. Code Quality and Testing
9. Release Management

2.2 Key Aspects - Details 1/3

Version Control:

- Version control systems (VCS) like Git are central to managing a codebase. These tools track changes made to the code over time, allowing developers to revert to earlier versions, branch out to work on different features, and merge changes when necessary.

Branching and Merging:

- Each developer works on their own branch and makes changes without affecting the main codebase.
- Merging allows changes from different branches to be integrated back into the main.
- Pull requests are often used for reviewing and integrating changes.

Code Review:

- Codebase management encourages thorough code reviews before changes are merged. This ensures that new code is high quality, follows project conventions, and does not introduce bugs or vulnerabilities.

2.2 Key Aspects - Details 2/3

Dependency Management:

Most software projects depend on external libraries or packages. Codebase management includes managing these dependencies, ensuring that the correct versions are used, and avoiding conflicts or compatibility issues.

Continuous Integration and Continuous Deployment (CI/CD):

In modern codebase management, CI/CD pipelines are often implemented to automatically test, build, and deploy the code whenever changes are made. This ensures that the codebase remains stable, functional, and up-to-date.

Issue Tracking:

Codebase management is closely integrated with issue tracking systems (like Jira or GitHub Issues). These systems allow developers to associate tasks or bugs with specific code changes, ensuring that the project is progressing smoothly.

2.2 Key Aspects - Details 3/3

Documentation:

Proper documentation is part of good codebase management. This includes inline comments, external documentation (like READMEs or wikis), and other supporting materials to help developers understand how to work with the codebase.

Code Quality and Testing:

Ensuring high code quality through practices like automated testing (unit tests, integration tests) is a key aspect of managing a codebase. This helps prevent bugs and regressions and ensures that the code remains maintainable.

Release Management:

Codebase management also includes planning and controlling the release cycle, managing versioning, and handling deployment processes.

2.3 GitHub Codebase Management

"The use of GitHub to manage a project's codebase”

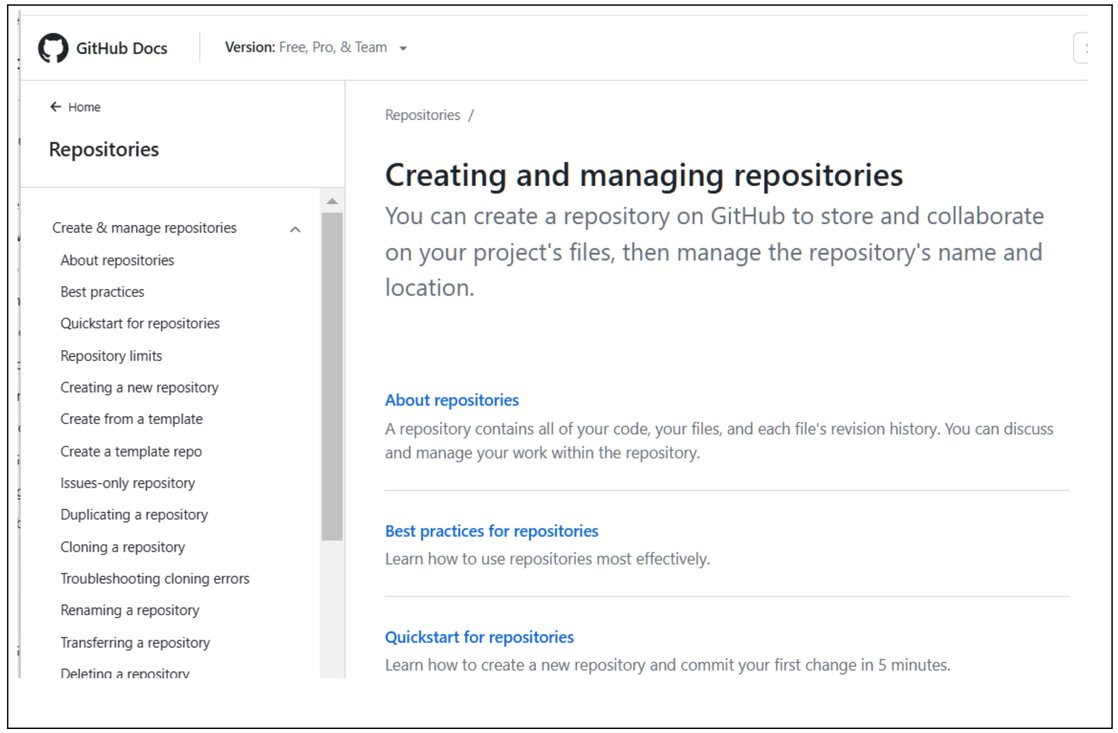
Aspects Manageable through GitHub:

- **Repository Organization:** Structuring of repositories for different projects and teams.
- **Workflows:** Clear processes for code development, review, and deployment.
- **Security:** Protecting the codebase from vulnerabilities and unauthorized access.
- **Automation:** Automating tasks like building, testing, and deploying code.
- **Integration:** Integrating of SCM with other development tools (e.g., CI/CD pipelines).

2.4 Creating and Managing Repositories

In-depth information about creating and managing repositories on GitHub.

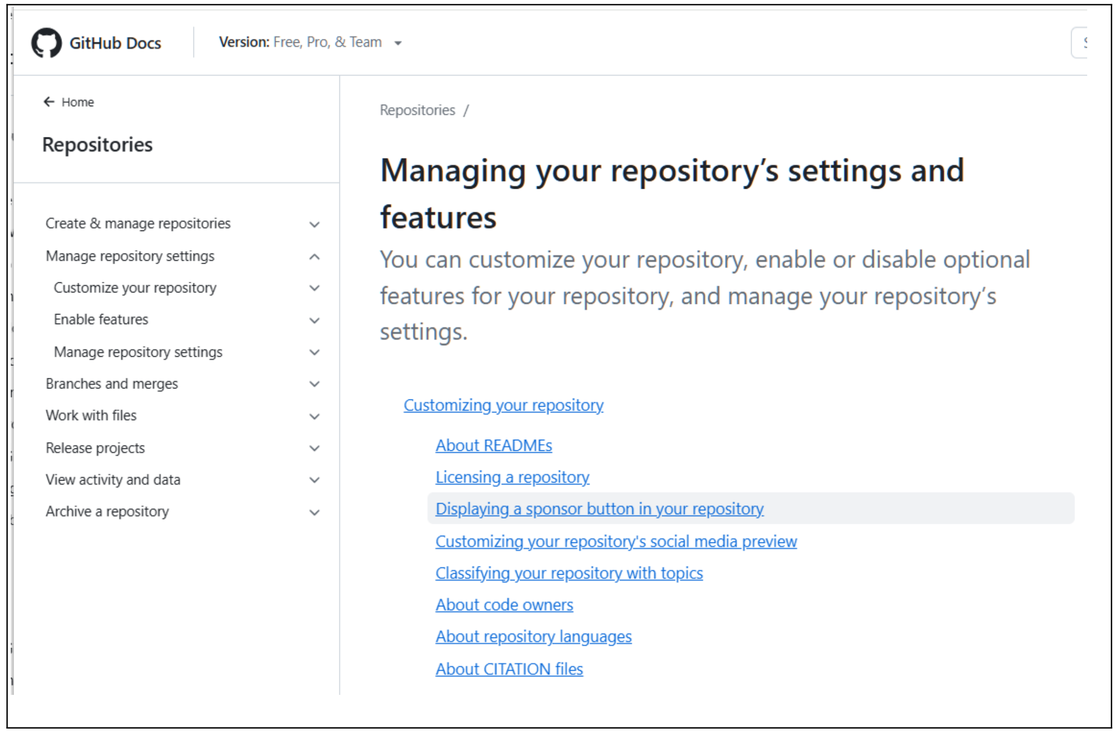
https://docs.github.com/en/repositories/creating-and-managing-repositories



2.5 Managing your Repository's Settings & Features

Covers setup of settings for repositories

https://docs.github.com/en/repositories/managing-your-repositorys-settings-and-features



2.6 Ten GitHub Best Practices for Repository Management

Video with practical advice for repository management.

https://www.youtube.com/watch?v=VvgoOgWFiZY