Category	Question	Yes	No	Evidence
Q1 - Software Overview				
Question 1.1	Does your website and documentation provide a clear, high-level overview of your software?			The WrikiCafe website and **README.md**, and DOCUMENTATION.md give a crisp, high-level snapshot of what the platform does and who it serves. The documentation clearly outlines purpose, scope, stakeholders, and differentiating features such as smart personalization, real-time order updates, and smooth UX.
Question 1.2	Does your website and documentation clearly describe the type of user who should use your software?			The wrikicafe documentation (Section 2: Intended Users) clearly identifies and describes the two main types of users who should use the software: Customers (Students & Staff): use the web app to browse an interactive café menu, receive smart AI-based recommendations, place and track orders, and get instant notifications when their drinks are ready. Administrators: manage and monitor orders through a dedicated dashboard, update order status (Pending → In Progress → Ready → Completed), curate menu items, and send real-time notifications to customers.
Question 1.3	Do you publish case studies to show how your software has been used by yourself and others?			While wrikicafe is primarily an academic project, the team has developed and documented internal case studies that demonstrate how the software is actively used and tested across multiple user roles: Admin Case Study: Showcases how administrators manage real café operations digitally — adding menu items, updating order statuses, and triggering real-time “Ready for Pickup” notifications. Screenshots and demo walkthroughs highlight how the Admin Dashboard streamlines order management. Customer Case Study: Demonstrates a student user placing an order through the menu interface, receiving AI-driven drink recommendations, and viewing live updates as their order progresses from “Pending” to “Ready.” End-to-End Demo: The documentation and project video act as a practical case study, showing the software in use from both admin and customer perspectives. These examples are supported by the live demo, README, and demo video link clearly illustrate how wrikicafe has been used by the project team to validate all major features and user flows.
Q2 - Identity				
Question 2.1	Is the name of your project/software unique?			Yes. It is called WrikiCafe. Wriki means she-wolf in sanskrit. The name combines “wriki” (a creative, memorable prefix) with “cafe,” reflecting both the team identity and the software’s focus on café-style ordering. A search across GitHub, NPM, and general web results shows no existing project or commercial product using the same name.
Question 2.2	Is your project/software name free from trademark violations?			Yes. The name “wrikicafe” is original, created for this project, and not associated with any existing brand or trademark.
Q3 - Availability				
Question 3.1	Is your software available as a package that can be deployed without building it?			We added an INSTALL.md file with clear setup and run instructions and updated the root package.json to enable one-command startup for both frontend and backend, making deployment simple and consistent without Docker. We have also hosted our website on render. Frontend: https://wrikicafe.onrender.com/ Backend: https://wrikicafe-backend.onrender.com/
Question 3.2	Is your software available for free?			Yes. It is an open-source project, code is present on a public repository on github. https://github.com/RishithaRamesh/wolfcafeplus
Question 3.3	Is your source code publicly available to download, either as a downloadable bundle or via access to a source code repository?			Yes. It is an open-source project, code is present on a public repository on github. https://github.com/RishithaRamesh/wolfcafeplus
Question 3.4	Is your software hosted in an established, third-party repository like GitHub, BitBucket, LaunchPad, or SourceForge?			Yes. We have hosted it on render. https://wrikicafe.onrender.com/
Q4 - Documentation				

Question 4.1	Is your documentation clearly available on your website or within your software?			The full documentation is included in the project's README.md and DOCUMENTATION.md, both linked directly from the website footer and the GitHub repository for easy access.
Question 4.2	Does your documentation include a "quick start" guide, that provides a short overview of how to use your software with some basic examples of use?			The INSTALL.md and README.md files include a Quick Start section outlining installation, setup, and basic usage steps—showing how users can clone the repo, install dependencies, and run wrikicafe with a single command.
Question 4.3	If you provide more extensive documentation, does this provide clear, step-by-step instructions on how to deploy and use your software?			The extended documentation (INSTALL.md and README.md) provides clear, step-by-step instructions for setting up, deploying, and using wrikicafe—covering environment setup, dependency installation, running the app locally, and accessing the live Render deployment.
Question 4.4	Do you provide a comprehensive guide to all your software's commands, functions and options?			`INSTALL.md and README.md`
Question 4.5	Do you provide troubleshooting information that describes the symptoms and step-by-step solutions for problems and error messages?			`INSTALL.md and README.md`
Question 4.6	If your software can be used as a library, package or service by other software, do you provide comprehensive API documentation?			A complete and publicly available API_DOCUMENTATION.md file has been generated, detailing all endpoints, methods, roles, authentication requirements, and example requests/responses—providing comprehensive API documentation for the wrikicafe service.
Question 4.7	Do you store your documentation under revision control with your source code?			All documentation files (README.md, DOCUMENTATION.md, and INSTALL.md) are version-controlled within the same GitHub repository as the source code, ensuring updates are tracked alongside software changes.
Question 4.8	Do you publish your release history e.g. release data, version numbers, key features of each release etc. on your web site or in your documentation?			The project's RELEASE_NOTES.md and GitHub Releases section document version numbers, release dates, and key feature updates for each version (e.g., v1.0.0, v1.1.0), providing a clear public release history.
Q5 - Support				
Question 5.1	Does your software describe how a user can get help with using your software?			The README.md and DOCUMENTATION.md include a Support & Help section that explains how users can get assistance—through the project's GitHub Issues page for bug reports or questions, and via detailed usage instructions in the documentation.
Question 5.2	Does your website and documentation describe what support, if any, you provide to users and developers?			The documentation specifies that community support is provided through the project's GitHub Issues for bug reports and feature requests, and that setup and usage guidance are available in the README.md and INSTALL.md for both users and developers.
Question 5.3	Does your project have an e-mail address or forum that is solely for supporting users?			Yes
Question 5.4	Are e-mails to your support e-mail address received by more than one person?			Yes
Question 5.5	Does your project have a ticketing system to manage bug reports and feature requests?			We have a kanban board on github - https://github.com/users/RishithaRamesh/projects/2
Question 5.6	Is your project's ticketing system publicly visible to your users, so they can view bug reports and feature requests?			Yes
Q6 - Maintainability				
Question 6.1	Is your software's architecture and design modular?			The software follows a modular MERN architecture — separating the frontend (React components and contexts), backend (Express controllers, routes, and models), and database (MongoDB with Mongoose schemas) — ensuring scalability, maintainability, and clear separation of concerns.
Question 6.2	Does your software use an accepted coding standard or convention?			The project follows standard JavaScript/React coding conventions enforced through ESLint and Prettier configurations, with automated GitHub Actions lint checks ensuring consistent, readable, and high-quality code across the frontend and backend.
Q7 - Open Standards				
Question 7.1	Does your software allow data to be imported and exported using open data formats?			The software uses JSON, an open and widely accepted data format, for all API requests and responses—allowing seamless data import, export, and integration with other systems or analytical tools.
Question 7.2	Does your software allow communications using open communications protocols?			The software uses Socket.IO, which is built on open web protocols (WebSockets and HTTP) to enable real-time, bidirectional communication between the client and server—supporting open, standardized communication.
Q8 - Portability				

Question 8.1	Is your software cross-platform compatible?			The application is entirely web-based, built with React (frontend) and Node.js/Express (backend), making it fully cross-platform—accessible on any device or operating system through a modern web browser.
Q9 - Accessibility				
Question 9.1	Does your software adhere to appropriate accessibility conventions or standards?			The frontend follows basic web accessibility practices—using semantic HTML, alt text for images, clear color contrast, and responsive design with Tailwind CSS—to ensure usability across diverse users and devices.
Question 9.2	Does your documentation adhere to appropriate accessibility conventions or standards?			The documentation is written in clear, structured Markdown with descriptive headings, bullet points, and accessible text formatting—ensuring readability through screen readers and compatibility with GitHub's built-in accessibility features.
Q10 - Source Code Management				
Question 10.1	Is your source code stored in a repository under revision control?			All source code is maintained in a GitHub repository using Git for full revision control, allowing version tracking, branching, and collaborative development.
Question 10.2	Is each source code release a snapshot of the repository?			Each versioned release on GitHub (e.g., v1.0.0) is tagged and published as a snapshot of the repository at that point in time, preserving the exact source code and documentation for that release.
Question 10.3	Are releases tagged in the repository?			The repository includes two tagged releases (e.g., v1.0.0 and v1.1.0), each marking a stable snapshot of the project's source code and features at the time of release.
Question 10.4	Is there a branch of the repository that is always stable? (i.e. tests always pass, code always builds successfully)			The main branch is maintained as the stable version of the code, with all tests passing and successful builds. New features and bug fixes are developed in separate release and bugFix branches before being merged into main.
Question 10.5	Do you back-up your repository?			A scheduled GitHub Action (backup.yml) automatically backs up the repository every week, ensuring all code and documentation are securely archived.
Q11 - Building & Installing				
Question 11.1	Do you provide publicly-available instructions for building your software from the source code?			The INSTALL.md file provides clear, publicly available instructions for cloning the repository, installing dependencies, and running both the frontend and backend from source using simple npm commands.
Question 11.2	Can you build, or package, your software using an automated tool?			The software uses npm scripts and GitHub Actions for automated builds—npm run build compiles the frontend, and CI workflows automatically build and test the project on every push or pull request.
Question 11.3	Do you provide publicly-available instructions for deploying your software?			The INSTALL.md file includes a detailed Deployment Guide with step-by-step instructions for deploying the frontend and backend on Render, making the process fully reproducible and publicly accessible.
Question 11.4	Does your documentation list all third-party dependencies?			The documentation and package.json files in both the frontend and backend clearly list all third-party dependencies—such as React, Express, MongoDB, Axios, and Tailwind CSS—ensuring full transparency of external libraries used.
Question 11.5	Does your documentation list the version number for all third-party dependencies?			The exact version numbers of all third-party dependencies are specified in the package.json files for both the frontend and backend, allowing users to reproduce the same environment reliably.
Question 11.6	Does your software list the web address, and licences for all third-party dependencies and say whether the dependencies are mandatory or optional?			The project's LICENSE.md and documentation reference all third-party dependencies with their respective licenses and npm source URLs, as listed in the package.json files, and indicate which libraries are mandatory (core runtime) versus optional (development or styling tools).
Question 11.7	Can you download dependencies using a dependency management tool or package manager?			All dependencies can be automatically installed using npm, the Node.js package manager, by running npm install in the frontend and backend directories, ensuring consistent and easy setup.
Question 11.8	Do you have tests that can be run after your software has been built or deployed to show whether the build or deployment has been successful?			The project includes Jest and Supertest-based automated tests for backend controllers and models, which are executed after each build through GitHub Actions, verifying that the software functions correctly after deployment.
Q12 - Testing				
Question 12.1	Do you have an automated test suite for your software?			The project includes an automated Jest test suite integrated with GitHub Actions, covering key backend components such as controllers, routes, and models to ensure reliability and continuous verification.
Question 12.2	Do you have a framework to periodically (e.g. nightly) run your tests on the latest version of the source code?			A scheduled GitHub Actions workflow is configured to automatically run the Jest test suite on the latest source code at regular intervals, ensuring continuous validation of build stability and functionality.
Question 12.3	Do you use continuous integration, automatically running tests whenever changes are made to your source code?			The project uses GitHub Actions for continuous integration, automatically running build and Jest test workflows on every push and pull request to verify that new changes don't break existing functionality.
Question 12.4	Are your test results publicly visible?			Test results are publicly visible through the project's GitHub Actions tab, where build and test workflow statuses (/) and detailed logs can be viewed for every commit and pull request.
Question 12.5	Are all manually-run tests documented?			The TESTING.md section in the documentation outlines the manually-run tests—such as verifying order updates, menu management, and notification flows—along with step-by-step procedures and expected outcomes.
Q13 - Community Engagement				

Question 13.1	Does your project have resources (e.g. blog, Twitter, RSS feed, Facebook page, wiki, mailing list) that are regularly updated with information about your software?			Not yet, since our project is fairly new. We will start working on it.
Question 13.2	Does your website state how many projects and users are associated with your project?			The Admin Dashboard displays live analytics, including the total number of users, orders, and top-selling menu items—providing clear visibility into project usage and engagement metrics.
Question 13.3	Do you provide success stories on your website?			The project's About Us page highlights the team's journey, goals, and real-world impact of wrikicafe—serving as a success story that showcases its development and usefulness within a campus café setting.
Question 13.4	Do you list your important partners and collaborators on your website?			The About Us page lists key collaborators and contributors, including the project team members Rishitha Ramesh, Rujuta Budke, and Dhruva Kamble, acknowledging their roles in developing and maintaining wrikicafe.
Question 13.5	Do you list your project's publications on your website or link to a resource where these are available?			The website's About Us and Documentation sections link to the project's GitHub repository and related academic documentation, serving as the central resource for all project publications and technical materials.
Question 13.6	Do you list third-party publications that refer to your software on your website or link to a resource where these are available?			The project currently does not have any third-party publications or external references; all available documentation and materials are produced internally by the wrikicafe development team.
Question 13.7	Can users subscribe to notifications to changes to your source code repository?			Users can watch or star the GitHub repository to receive automatic notifications about commits, releases, and issues, enabling real-time updates on source code changes.
Question 13.8	If your software is developed as an open source project (and, not just a project developing open source software), do you have a governance model?			As an open-source project, wrikicafe follows a clear governance model defined in its CONTRIBUTING.md and CODE_OF_CONDUCT.md, outlining contributor roles, review processes, and decision-making procedures for maintaining transparency and collaboration.
Q14 - Contributions				
Question 14.1	Do you accept contributions (e.g. bug fixes, enhancements, documentation updates, tutorials) from people who are not part of your project?			The project welcomes external contributions through GitHub pull requests, with clear contribution guidelines provided in the CONTRIBUTING.md file for bug fixes, feature enhancements, and documentation improvements.
Question 14.2	Do you have a contributions policy?			A detailed CONTRIBUTING.md file outlines the project's contribution policy, including coding standards, branching strategy, and review process for submitting bug fixes, enhancements, and documentation updates.
Question 14.3	Is your contributions' policy publicly available?			The CONTRIBUTING.md file is publicly accessible in the project's GitHub repository, allowing anyone to view the contribution guidelines and participate in development.
Question 14.4	Do contributors keep the copyright/IP of their contributions?			Under the MIT License in LICENSE.md, contributors retain the copyright and intellectual property rights to their individual contributions while granting permission for their code to be used and redistributed under the same open-source terms.
Q15 - Licensing				
Question 15.1	Does your website and documentation clearly state the copyright owners of your software and documentation?			The LICENSE.md and project footer clearly list Rishitha Ramesh, Rujuta Budke, and Dhruva Kamble (2025) as the copyright owners of the wrikicafe software and its documentation.
Question 15.2	Does each of your source code files include a copyright statement?			// Licensed under the MIT License. See LICENSE.md for details. // Licensed under the MIT License. See LICENSE.md for details.
Question 15.3	Does your website and documentation clearly state the licence of your software?			The LICENSE.md file and the README.md both clearly state that the software is released under the MIT License, with a summary and link provided for easy reference.
Question 15.4	Is your software released under an open source licence?			Evidence: The software is released under the MIT License, an approved open-source license that permits use, modification, and redistribution with proper attribution.
Question 15.5	Is your software released under an OSI-approved open-source licence?			The project uses the MIT License, which is an OSI-approved open-source license, ensuring full compliance with open-source standards and permissions.
Question 15.6	Does each of your source code files include a licence header?			// Licensed under the MIT License. See LICENSE.md for details. // Licensed under the MIT License. See LICENSE.md for details.
Question 15.7	Do you have a recommended citation for your software?			https://zenodo.org/records/17538135
Q16 - Future Plans				
Question 16.1	Does your website or documentation include a project roadmap (a list of project and development milestones for the next 3, 6 and 12 months)?			The project's README.md and ROADMAP.md include a clear development roadmap outlining planned milestones for the next 3, 6, and 12 months, detailing upcoming features such as staff view, AI personalization, analytics upgrades, and mobile app expansion.
Question 16.2	Does your website or documentation describe how your project is funded, and the period over which funding is guaranteed?			No funding received yet.
Question 16.3	Do you make timely announcements of the deprecation of components, APIs, etc.?			Deprecations and major API changes are communicated through GitHub Releases and updates in the RELEASE_NOTES.md, ensuring users are informed ahead of time about any component or endpoint changes.