In detail about activity tracking story

Mention bug blitz : say u could understand the flow of code base which helped me solve many issues in there tickets were pending so every quarter last week use to focus on bugs and issues

Part of hackathon winning team idea was of My team lead to reduce the In ITP application

WHAT DO WALMART WANT

->WHAT U DID AND WHAT U KNOW NOT WHAT IS THE PRODUCT…  
-> GOOD DSA

* DEBUGGING SKILLS
* CODE BASE EASILY NAVIGATE
* PROOF OF WORK WON BUG BLITZ
* CROSS TEAM FUNCTIONALITY, I HAVE TO WORK WITH TESTING TEAM AND PRODUCTION TEAM SINCE THEY WORK ON RELASE.PY GROUP WITH FRONTDOOR TEAM HAD TO WORK ON DAYLIGHT TIMING TOO
* Mention about refdata controller and how flow cmes to refdata service through api
* How u used to debug effectively
* With api endpoint search and navigate
* Java lambda functions
* Cover all test cases
* Assert Junit test case
* Router test case
* \*\*\*\*Mention about the Nike issue
* Mention about Wellsfargo issue
* The db division
* Apple s some division they wont compromise on security or any other thing it was quite difficult for our team to satisfy them

1.WHY YHOUR NEXT JS APP IS SCALABLE?  
To make a Next.js application scalable, you can follow these best practices:

1. \*\*Use a scalable folder structure\*\*: Implement a well-organized folder structure that separates components, utilities, styles, and static assets into different folders. This approach improves navigation, reusability, scalability, and maintenance. For example, you can create a `components` folder for reusable components, a `utils` folder for utility functions, a `styles` folder for global and theme styles, and a `public` folder for static assets[2].

2. \*\*Leverage Next.js features\*\*: Next.js offers several features that can help improve scalability, such as automatic code splitting, optimized performance, and server-side rendering. These features can help reduce the load time of your application and improve the user experience[1][2].

3. \*\*Implement API routes\*\*: Use Next.js API routes to handle server-side logic and data fetching. This approach can help improve scalability by separating the front-end and back-end code and allowing for easier maintenance and updates[1][2].

4. \*\*Use a monorepo structure\*\*: Implement a monorepo structure using tools like TurboRepo, which can help distribute tasks across the dependency graph and improve scalability. This approach can help reduce the build time of your application and improve the developer experience[3].

5. \*\*Implement a scalable architecture\*\*: Use a scalable architecture that can handle large amounts of traffic and data. This approach can help ensure that your application can handle future growth and maintain performance[4].

6. \*\*Use a production-ready template\*\*: Use a production-ready and scalable template for building Next.js applications. This approach can help ensure that your application is built with best practices and is ready for production[5].

By following these best practices, you can improve the scalability of your Next.js application and ensure that it can handle future growth and maintain performance.