How can security and AI be leveraged to address modern security challenges—especially those found in web applications?
Participants are encouraged to:

- Investigate and identify potential vulnerabilities in existing web applications.
- Resolve vulnerabilities or use them to demonstrate enhanced security features.
- Propose or build security features that use AI or other techniques.
- Leverage OpenAI if you choose to, but we do ask that you only use free versions to maintain fairness.

Dates & Timing

- **Registration Note:** Though the first day of the event starts February 27th at 9am, you are more than welcome to register late and participate! The latest you can register is the second day Feburry 28th, before 12:00pm.
- **Submission Period:** February 28th 5:00pm to 6:00pm, 2025
- **Judging Period:** March 1 - March 2nd, 2025
- **Winners Announced:** March 3rd, 2025

Eligibility to Compete

- **Open to all undergraduate students and high school students.**
    - We ask all high school students to understand that they are going up against undergraduate students who specialize in these career fields so this will be tough!
- **Teams of 3 members** are required.

NON-eigibility to Compete Include the following:

- Employees, contractors, consultants, or agents affiliated with the Sponsor or Administrator in the last year.
- Employees of entities in which the Sponsor has ownership interest.
- Government employees or officials.
- The following can attend the event to see the competitors but NOT COMPETE.

How To Enter

Visit **https://bokohacks.com/** to register your team and submit your project.

Project Requirements

1. **What to Create**
    - Projects should address the overarching question of using **AI** and **cybersecurity** to solve modern security challenges in **web applications**.
    - Emphasize identifying and/or fixing **vulnerabilities** and integrating **innovative security features**.
1. **Functionality**
    - The project must be **functional and runnable** when being submitted.
1. **Platforms**
    - Ensure compatibility with any specified platforms or frameworks stated in the submission requirements (e.g., web-based solutions should be browser-accessible or containerized, if required).
1. **New Work Only**
    - All coding must be done during the hackathon.
    - Research, planning, and conceptualization beforehand are allowed.

1. **Testing**
    - Participants should provide clear ways for judges to **test** or **evaluate** vulnerabilities, fixes, or AI-based tools during the judging period.

Submission Requirements

1. **Working Project + README (2)**
    - Provide a README that includes:
        - **Team member names**
        - **Tools and resources used**
        - **Build and run instructions** for the project
1. **Demonstration Video (Max 3 minutes)**
    - Highlight key features, especially any **vulnerability scanning** or **security enhancement** modules.
    - Show how an attacker might exploit a web app, and/or how your solution defends against such exploits.
    - Upload to YouTube, Vimeo, or any similar platform, and provide the link.
1. **Code Repository**
    - Public GitHub repository (or another platform) link.
    - Must show **commit history** that aligns with hackathon timing rules.
1. **API/Tools Used**
    - List all **external APIs, AI frameworks, or security tools** integrated into your project
1. **Future Improvements**
    - Outline potential expansions (e.g., broader vulnerability coverage, additional AI models, refined UI/UX, etc.).
1. **Working Demo**
    - Provide access to a live demo, test build, or website (if relevant).
    - If credentials are needed, include them.
1. **Originality**
    - Projects must be original and respect **intellectual property rights**.

Ideas & Directions for Participants

Below are some potential project ideas that focus on both **finding vulnerabilities** in web applications and **enhancing** security. Feel free to pass these along to participants for inspiration:

1. **Improved captcha generation**
    - Fed up with bots spamming your forms? Try creating a more user-friendly captcha that still keeps automated attacks at bay.
1. **Secure Logging & Monitoring**
    - Set up detailed logs for critical activities (like logins or admin actions), and build a simple monitoring tool that flags anything suspicious.
1. **Multi-Factor Authentication**
    - Add an extra step (like a text code or an authenticator app) to your login process. It's not as scary to implement as you might think, and it's a huge security win.
1. **Enhance Current Features with Security in Mind**
    - Got an existing feature that's a bit wobbly on security? Shore it up by reviewing how data is handled or adding checks for risky user inputs.
1. **Demonstrate Web Based Security**
    - Want to show off your security know-how? Build a small demo that clearly illustrates a common attack (like XSS or SQL injection) and then show how to prevent it.
1. **Not Sure? No worries!**
    - If you're stuck or not feeling any of these ideas, just look for any security holes in the app and fix them. Even small patches can make a big difference.
1. **Privacy Enhancement via Secure Data Handling**

    - Focus on encrypting or anonymizing sensitive data, like user info or transaction records. This is especially helpful if your project deals with personal data.
1. **Fun and innovative way to make the website more secure**
    - Get creative—maybe turn security testing into a mini-game or add a humorous twist while still reinforcing best practices. It's a great way to keep folks engaged.

Tips for Success

- **Start Small and Prioritize**
  Don't stress about trying to fix every single thing from day one. Zero in on the most critical security gaps first, then tackle the smaller ones if you have time.
- **Test as You Go**
  After you implement a fix or a new feature, do a quick test before moving on. It's way easier to catch issues early than to dig through everything at the end.
- **Plan Your Time**
  With only 8 hours of coding, keep the scope realistic. Prove out a core feature (like scanning or patch recommendation) rather than trying to tackle everything at once.
- **Ask For Feedback**
  Whether it's a teammate, a classmate, or an online community, extra eyes on your code or approach can spot blind spots you might have missed. Plus, collaboration can spark new ideas!
- **Document Your Steps**

  You don't need a fancy write-up. Just jot down what you changed, why you changed it, and how to test it. It helps you keep track of your progress, and it's super handy if you come back later or pass it along to someone else.

- **Stay Curious**

  Security is always evolving. If you see something you don't fully understand, take a quick detour to look it up. You might discover new tools or techniques that save you time in the long run.

- **Show Off What You Did**

  At the end of the day (or project!), share your results—whether it's a demo video, a short presentation, or a quick walkthrough. It helps you solidify what you've learned and gives others a chance to learn from your work too.