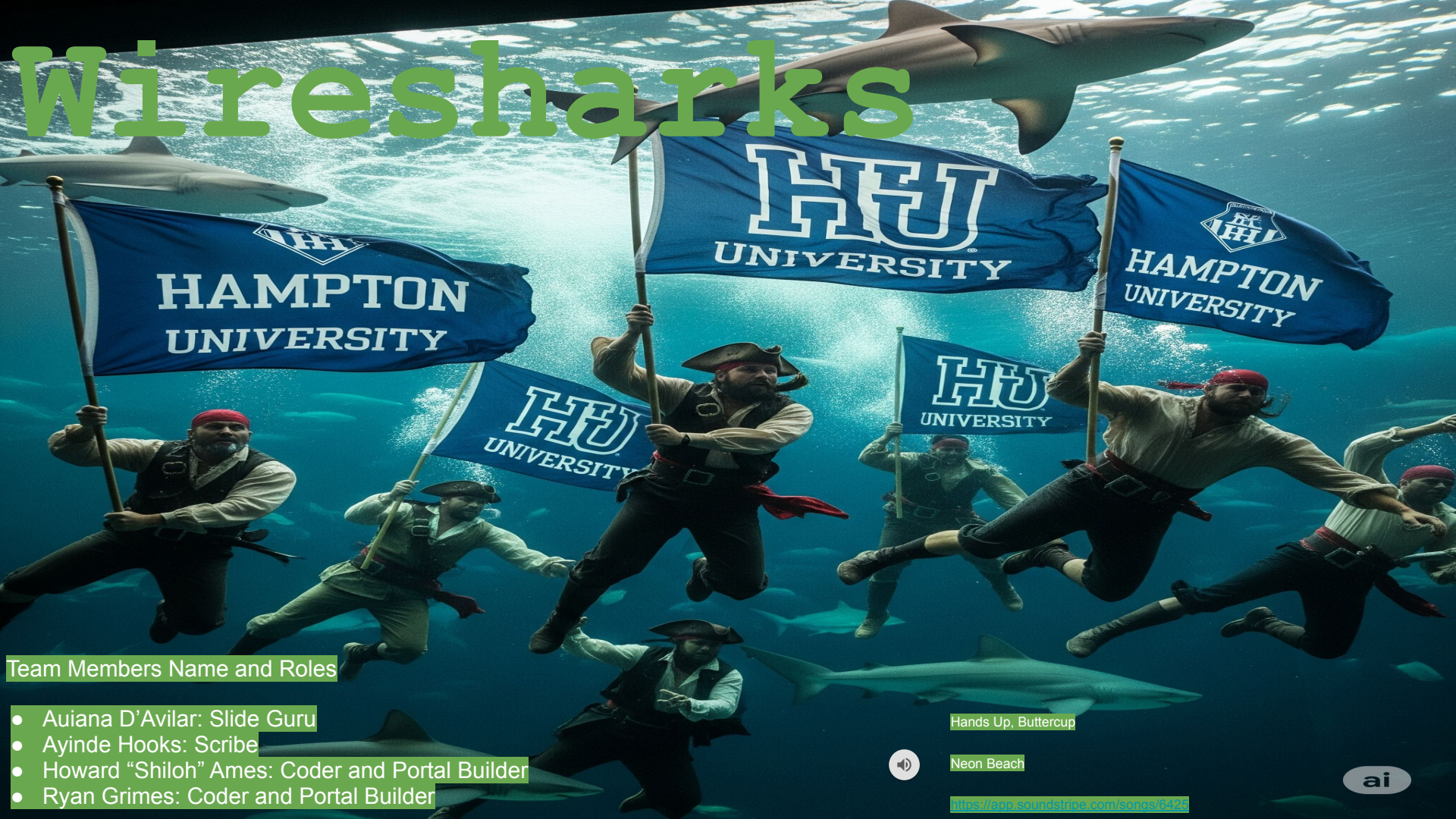
Wiresharks



Team Members Name and Roles

- Auiana D'Avilar: Slide Guru
- Ayinde Hooks: Scribe
- Howard "Shiloh" Ames: Coder and Portal Builder
- Ryan Grimes: Coder and Portal Builder

Hands Up, Buttercup

Neon Beach

https://app.soundstripe.com/songs/6425

Meet the Sharks

Ayinde Hooks

Scribe

Ryan Grimes

Coder and Portal Builder

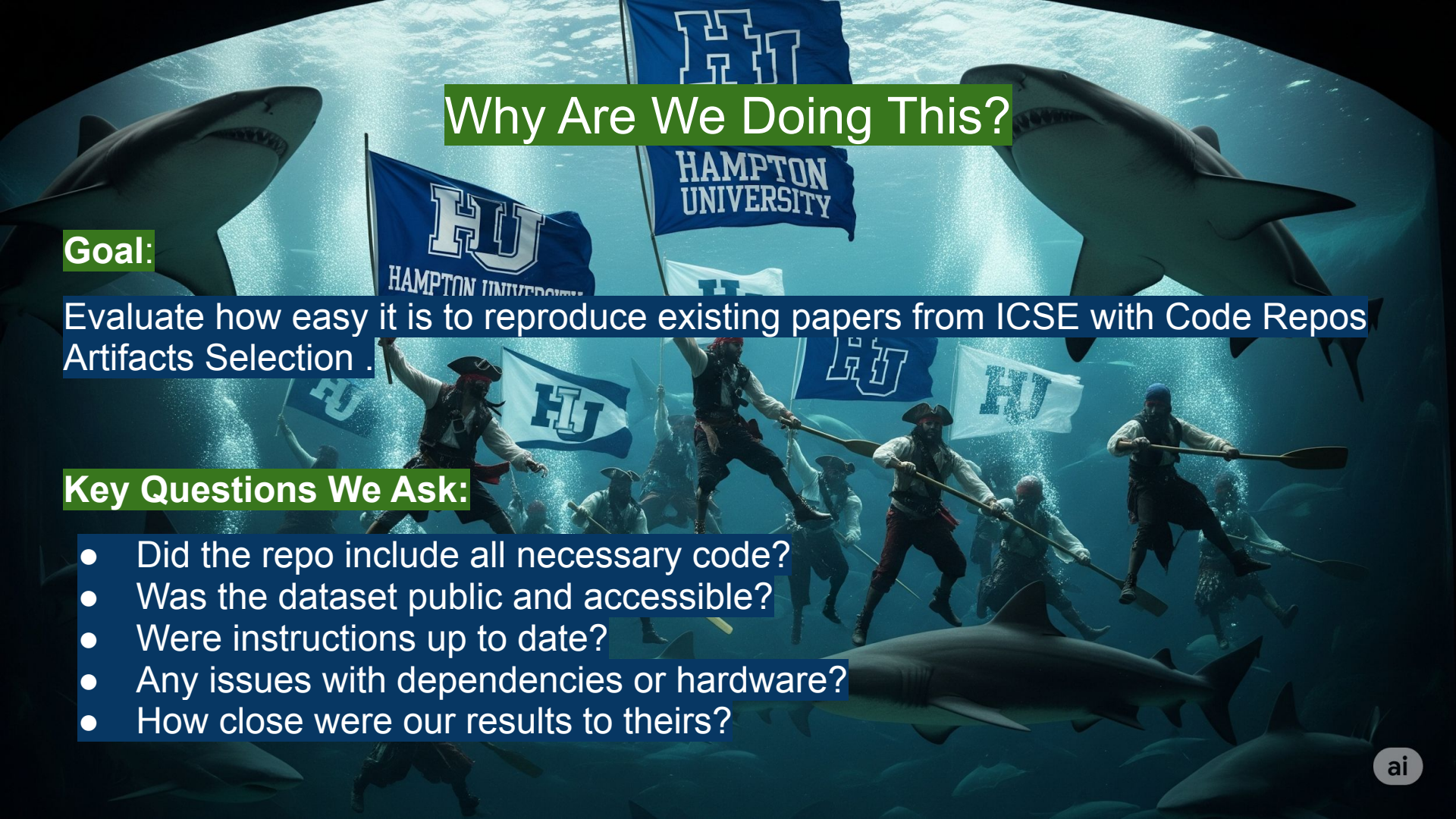
Howard “Shiloh” Ames

Coder and Portal Builder

Auiana D’Avilar

Slide Guru

Why Are We Doing This?



Goal:

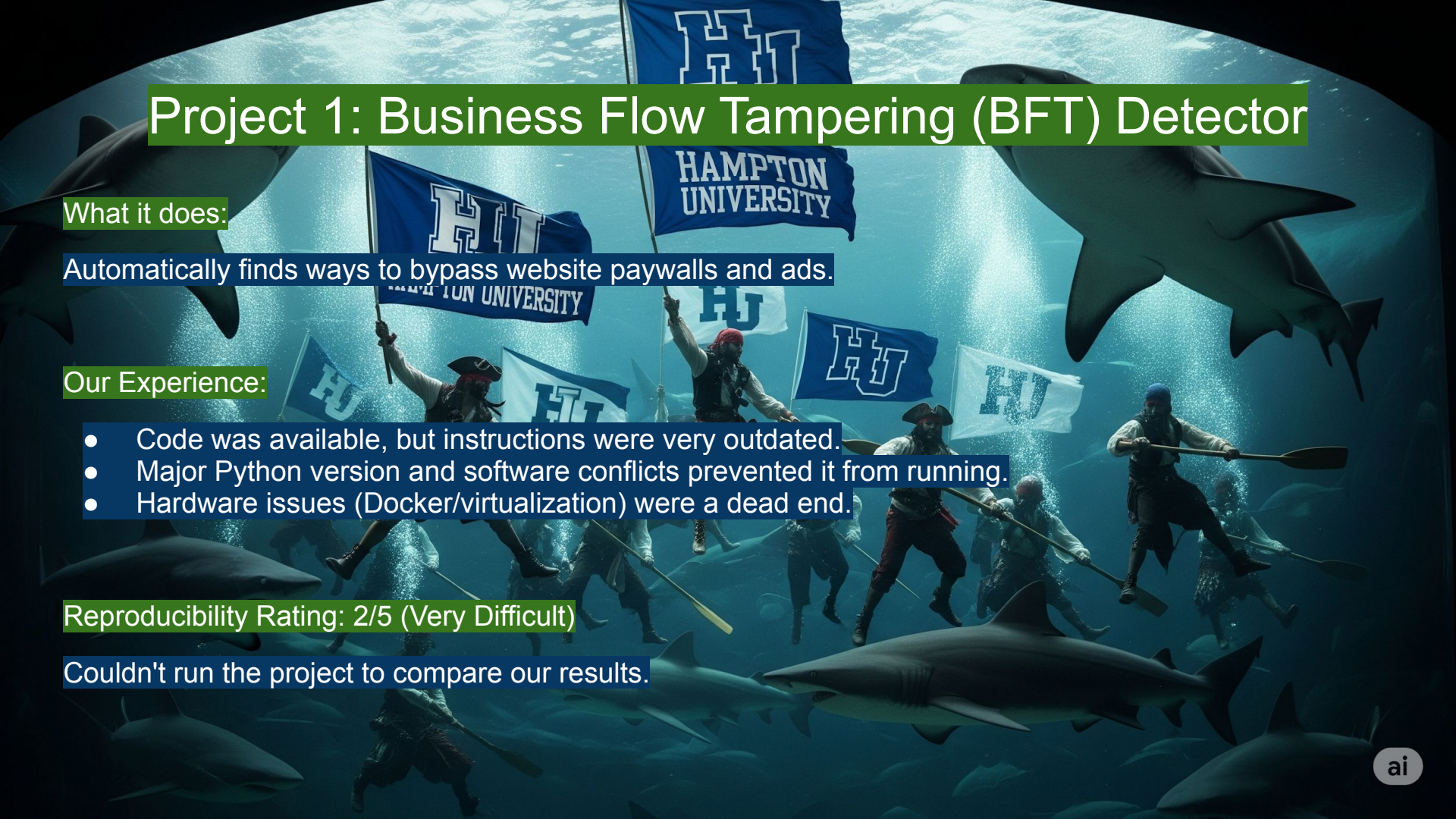
Evaluate how easy it is to reproduce existing papers from ICSE with Code Repos Artifacts Selection .

Key Questions We Ask:

- Did the repo include all necessary code?
- Was the dataset public and accessible?
- Were instructions up to date?
- Any issues with dependencies or hardware?
- How close were our results to theirs?

Score Card

Paper Availability	Availability of Code and Software	Availability of Datasets	Computer Requirements	GPU Requirements	Documentation Quality	Ease of Setup	Reproducibility of Results	Rating
1: Unavailable/Impossible to find. 2: Paywalled/Very hard to access. 3: Available, but via obscure link. 4: Open-access, direct link. 5:Open-access, easily searchable.	1: No code/Private repository. 2: Code available, but major parts missing/broken. 3: Code available, incomplete/needs big fixes. 4: Code available, mostly complete, minor issues. 5: Code fully available, complete.	1: dataset missing. 2: Mentioned, but completely inaccessible. 3: Available, but very hard to find/access. 4: Available, but metadata is poor/incomplete 5: Fully accessible with complete metadata.	1: No info/Impossible to meet. 2: Vague/Requires rare hardware. 3: Specific, but hard to meet. 4: Specific, but common hardware. 5: Clear, common, and flexible hardware.	1: No info/Mandatory custom GPU. 2: GPU required, vague/high-end specs. 3: GPU optional, specific specs. 4: GPU optional, common specs. 5: No GPU required (CPU-only).	1: None or misleading. 2: Very poor 3: Basic or needs much interpretation. 4: Clear but few details. 5: Comprehensive & clear.	1: Cannot be run due to critical issues or missing parts. 2: Can't run without major problems; needs expert help or significant workarounds. 3: Can be run with some effort; requires troubleshooting or minor fixes. 4: Runs well with minimal effort; minor adjustments might be needed. 5: Runs perfectly by simply following the instructions; no issues.	1: Cannot be run due to critical issues or missing parts. 2: Can't run without major problems; needs expert help or significant workarounds. 3: Can be run with some effort; requires troubleshooting or minor fixes. 4: Runs well with minimal effort; minor adjustments might be needed. 5: Runs perfectly by simply following the instructions; no issues.	1 (Impossible): Cannot be run due to critical issues or missing parts. 2 (Very Difficult): Can't run without major problems; needs expert help or significant workarounds. 3 (Doable): Can be run with some effort; requires troubleshooting or minor fixes. 4 (Mostly Smooth): Runs well with minimal effort; minor adjustments might be needed. 5 (Plug and Play): Runs perfectly by simply following the instructions; no issues.



Project 1: Business Flow Tampering (BFT) Detector

What it does:

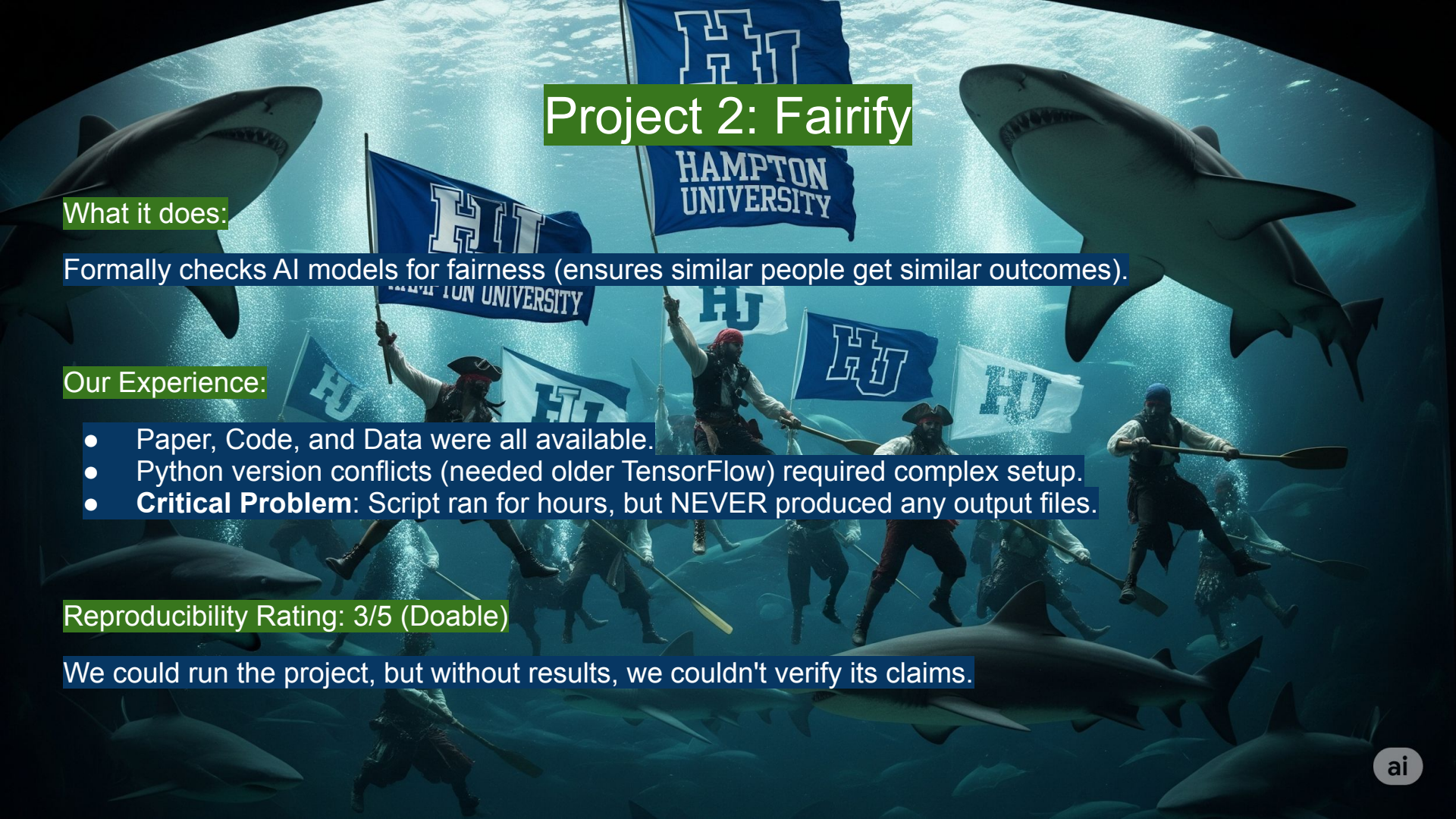
Automatically finds ways to bypass website paywalls and ads.

Our Experience:

- Code was available, but instructions were very outdated.
- Major Python version and software conflicts prevented it from running.
- Hardware issues (Docker/virtualization) were a dead end.

Reproducibility Rating: 2/5 (Very Difficult)

Couldn't run the project to compare our results.



Project 2: Fairify

What it does:

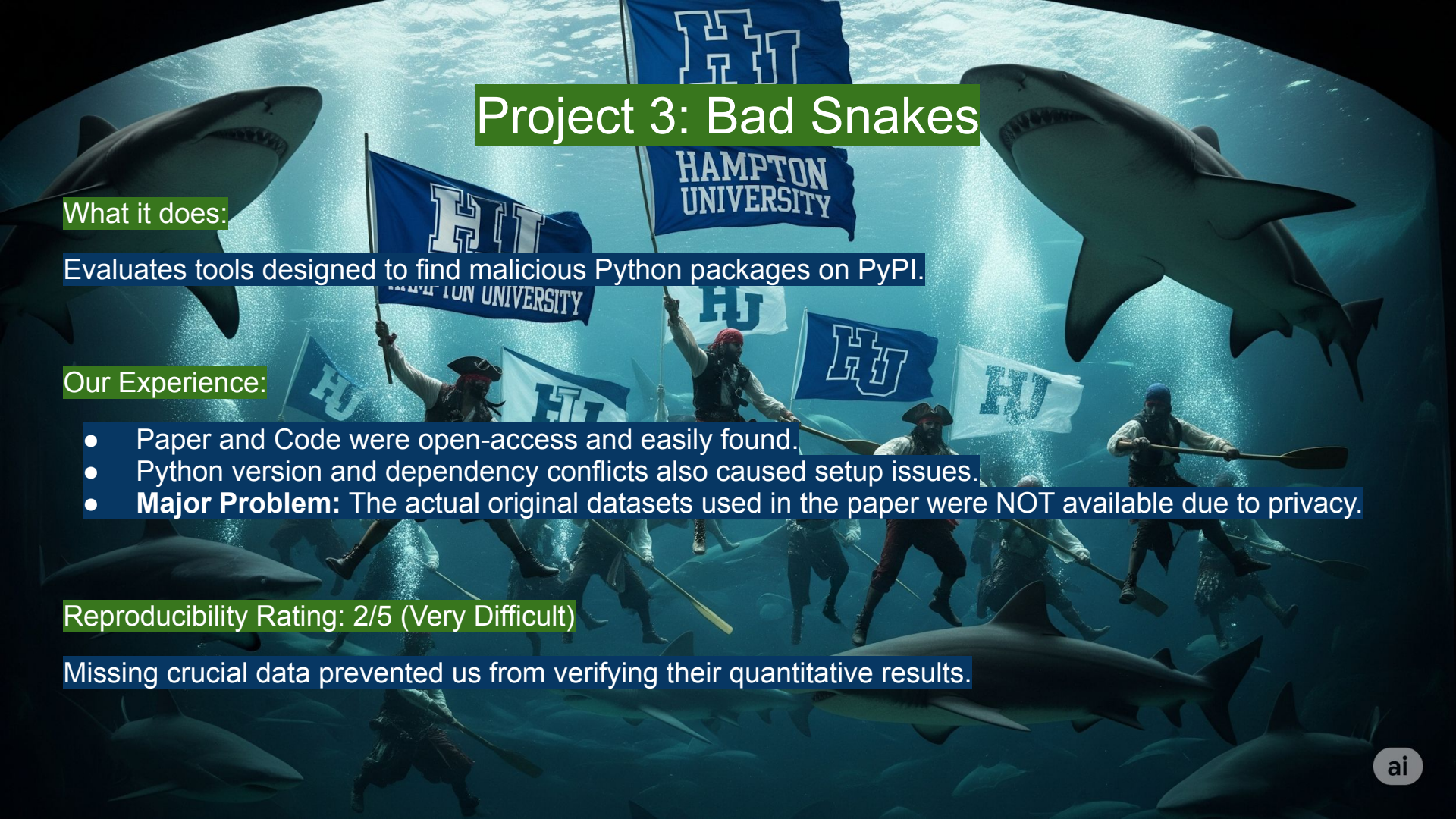
Formally checks AI models for fairness (ensures similar people get similar outcomes).

Our Experience:

- Paper, Code, and Data were all available.
- Python version conflicts (needed older TensorFlow) required complex setup.
- **Critical Problem**: Script ran for hours, but NEVER produced any output files.

Reproducibility Rating: 3/5 (Doable)

We could run the project, but without results, we couldn't verify its claims.



Project 3: Bad Snakes

What it does:

Evaluates tools designed to find malicious Python packages on PyPI.

Our Experience:

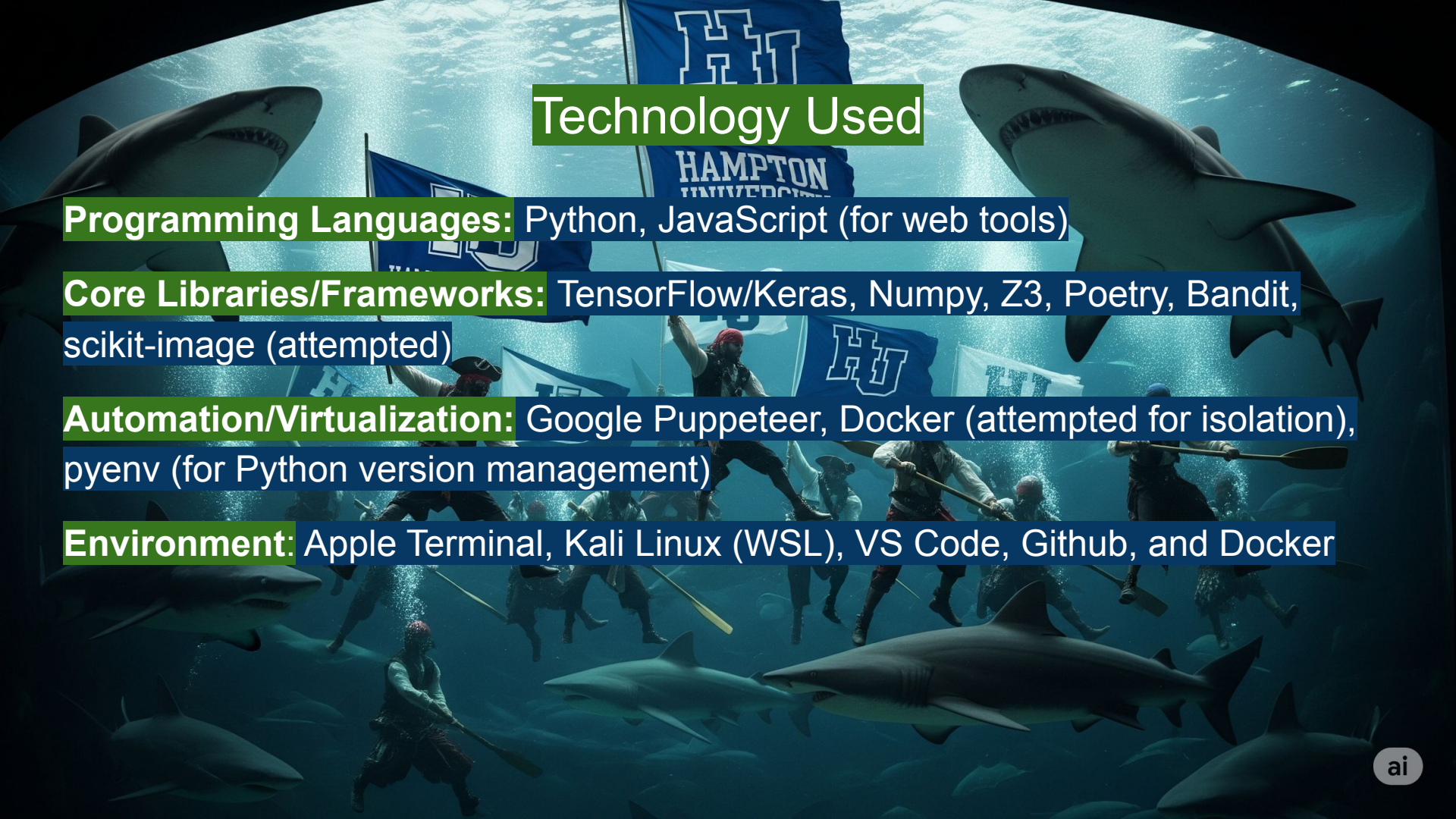
- Paper and Code were open-access and easily found.
- Python version and dependency conflicts also caused setup issues.
- **Major Problem:** The actual original datasets used in the paper were NOT available due to privacy.

Reproducibility Rating: 2/5 (Very Difficult)

Missing crucial data prevented us from verifying their quantitative results.

Common Challenges

- **Outdated Instructions**: Frequent source of setup failures.
- **Python Versioning**: Newer Python often breaks older project dependencies.
- **Missing Datasets**: A significant barrier to full reproduction.
- **Environment Specificity**: Unexpected hardware or system settings blocking progress.
- **Lack of Clear Output**: Scripts running but no results appearing.



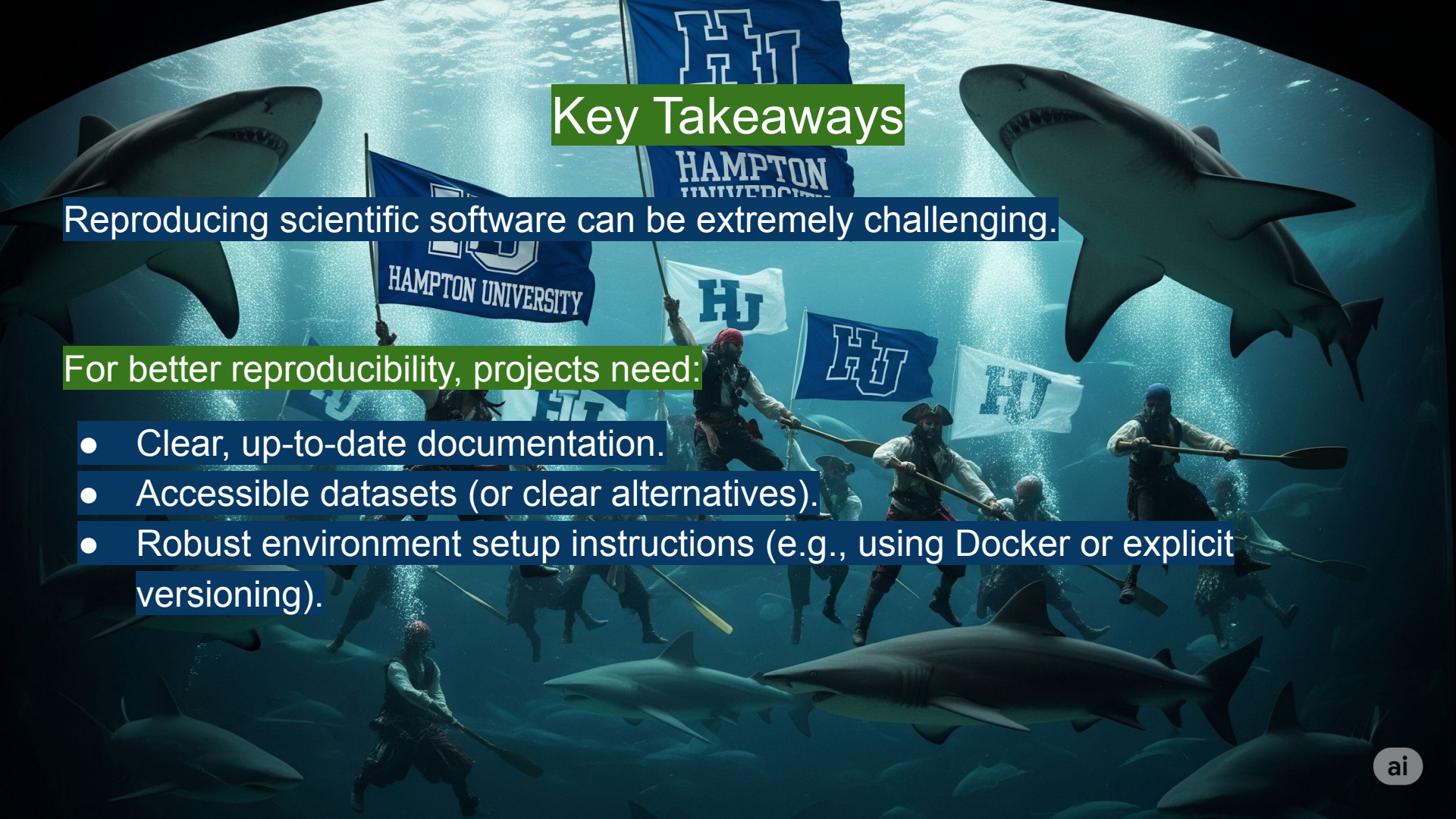
Technology Used

Programming Languages: Python, JavaScript (for web tools)

Core Libraries/Frameworks: TensorFlow/Keras, Numpy, Z3, Poetry, Bandit, scikit-image (attempted)

Automation/Virtualization: Google Puppeteer, Docker (attempted for isolation), pyenv (for Python version management)

Environment: Apple Terminal, Kali Linux (WSL), VS Code, Github, and Docker



Key Takeaways

Reproducing scientific software can be extremely challenging.

For better reproducibility, projects need:

- Clear, up-to-date documentation.
- Accessible datasets (or clear alternatives).
- Robust environment setup instructions (e.g., using Docker or explicit versioning).