Pair Programming Matcher

By Team ***Fifteam***: Laith Al-Masri, James Barber, Sidney Fredericks, Linley Spangler

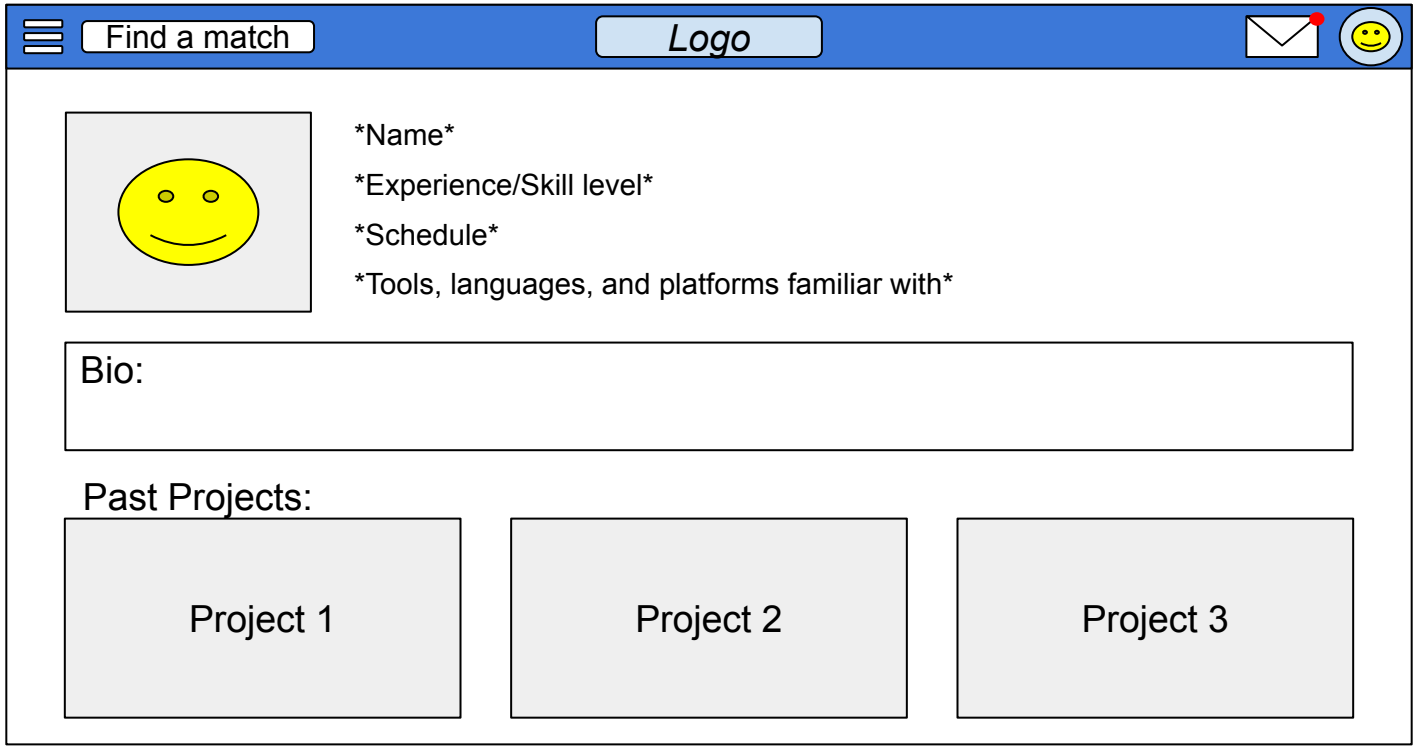
Problem Statement

- Difficulty finding skilled collaborators
  - Struggle to find programmers with desired technical skills needed for specific projects.
- Misaligned project goals
  - Face challenges due to differing project overall expectations or goals.
- Mismatch of experience levels
  - Programmers with different levels of experience may find it difficult to collaborate effectively.
- Lack of soft skills compatibility
  - Technical skills are not the only factor in successful collaborations.

Proposed Solution

- Our solution, the Pair Programming Matcher, addresses collaboration challenges by analyzing skill levels, preferences, and project requirements.
- It pairs programmers with complementary skills to ensure alignment with project goals, balances experience levels for effective teamwork, and incorporates soft skills compatibility for stronger interpersonal dynamics.
- This approach optimizes collaboration, enhances teamwork, and delivers higher-quality project outcomes.

Proposed Solution



Related Work

- Several articles that support the idea of pair programming
  - "Enhancing Teamwork and Collaboration: A Systematic Review of Algorithm-Supported Pedagogical Methods"
  - "11 Best Collaborative Coding Tools"
  - “Benefits of working in a Partnership: A model”
- There are several programs that do something similar to our project:
  - **Teammates** - Used by educators to form teams based on complementary skills and preferences
  - **Skill Display** - Platform for visualizing skill sets and matching individuals for collaboration
- **Bottom Line:** there is no platform that directly supports the goal that we have

Class Concepts Used

- **Agile Development - Throughout**
  - Used throughout the project to determine what was successfully completed and what needed to be worked on next
  - Useful for planning and project management
- **Requirements Elicitation - PM1**
  - Was done through the use of survey which gathered data from our peers about how they have tried to gain partners in the past
  - Useful for determining what users would want/need in our proposed solution

Class Concepts Used

- **Requirements Specification - PM2**
  - This was done through writing out user stories and was helpful for developing a well-rounded perspective on our application
  - Useful for understanding how application should function
- **Wireframing - PM3**
  - Was used to create a basic mock-up of our application and gain of visual understanding of it
  - Useful for planning out the user interface and gaining feedback

Limitations and Future Work

Limitations:

Skill and Compatibility Accuracy: While our matching algorithm is robust, its effectiveness depends heavily on the accuracy of user-provided data, such as skill levels and preferences.

Dynamic Changes in Projects: The tool may struggle to adapt if project requirements or team dynamics change significantly after initial matching.

Future Work:

Enhanced Data Collection: Implement advanced methods like skill-testing quizzes and behavioral surveys to improve the accuracy of skill and compatibility assessments.

AI-Driven Dynamic Matching: Introduce AI that continuously monitors team performance and suggests rematches or adjustments as needed.

Integration with Collaboration Tools: Seamlessly integrate with platforms like GitHub, Jira, and Slack for smoother workflow and data exchange.

Additional Features if We Had More Time

- **Advanced Analytics Dashboard:** Provide users and project managers with detailed analytics on team performance, collaboration patterns, and project outcomes.
- **Language and Cultural Matching:** Incorporate language preferences and cultural considerations for globally distributed teams.
- **Skill Development Suggestions:** Offer personalized learning paths or resource recommendations to help users improve skills relevant to their assigned projects.

What we learned

1. Gained a better understanding of what it looks like to develop a software application from start to finish
2. How to use different techniques to breakdown complex development projects into manageable parts
3. Continued to develop the skills that come with working as a part of a team
4. How to determine and analyze requirements for a software application
5. How to describe and propose software engineering solutions

Questions?