User Interface Development using CustomTkinter for Automating Python Dependency Recognition

Travis Chan, Dibyendu Brinto Bose, and Dr. Chris Brown
College of Engineering Department of Computer Science, Virginia Tech, Blacksburg, VA

INTRODUCTION

Timely recognition of and actionability toward resolving Python library dependency issues is crucial in supporting a productive software engineering workflow. Considering not all Python developers have machine learning (ML) knowledge, direct interaction with ML models utilized for backend development is not an ideal user experience. Thus, usability engineering plays a critical role in enabling Python developers with actionable information.

```
1  import pandas
2  import numpy
3  import matplotlib
```

Fig. 1: Python library import statements

✓ **Python libraries**
→ **Library dependencies**
✗ **Dependency issues**

METHODS

A. Understanding needs
a) State of the art tool interaction analysis

B. Designing solutions
a) "Emotional impact", "physical action", "existence of feedback"

C. Prototype candidates
a) User study prototype

D. Evaluate user experience
a) Nielsen's heuristics, user study

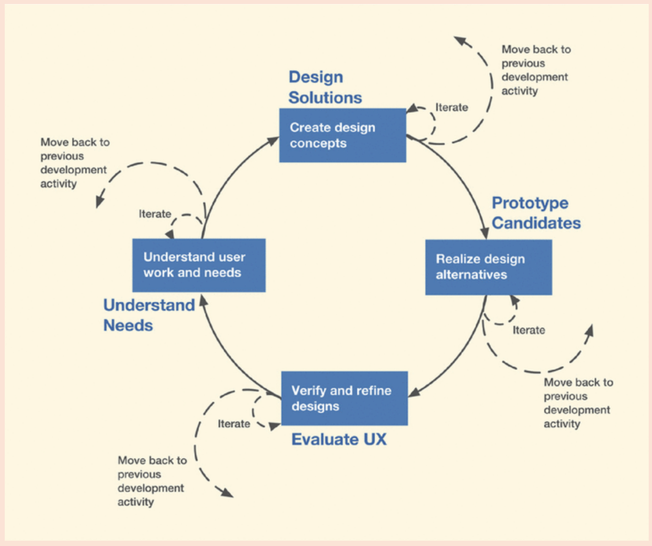
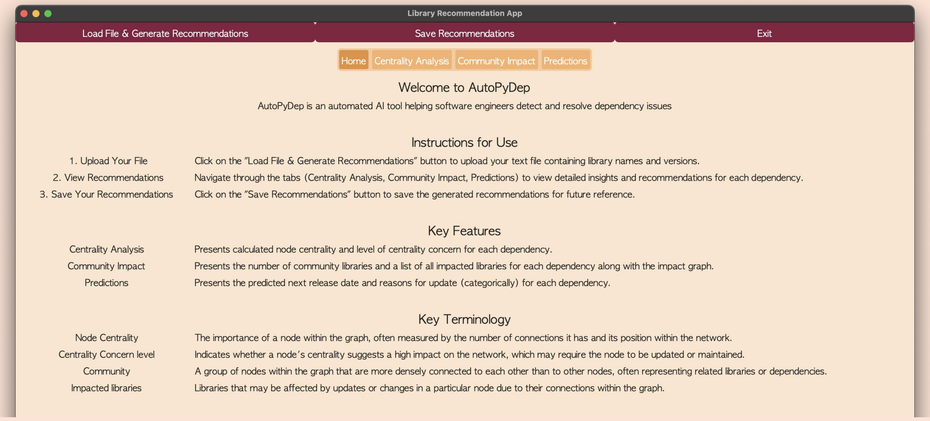
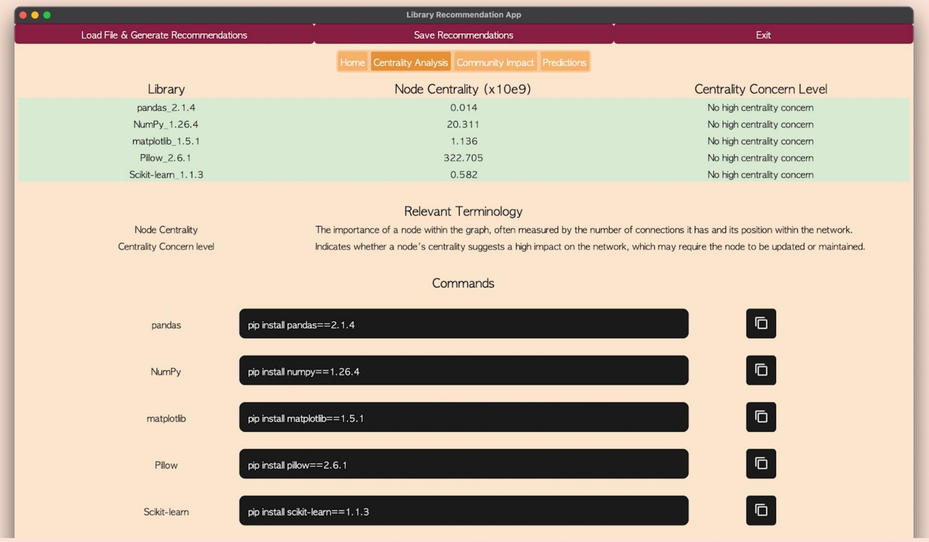
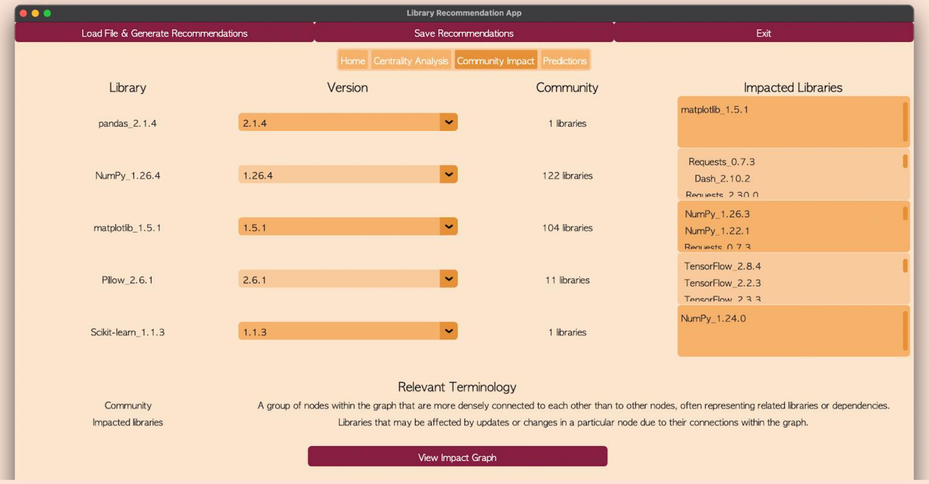


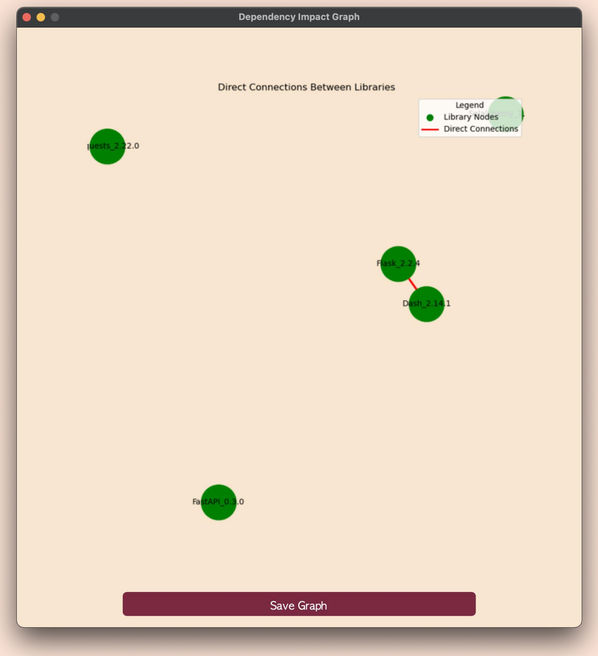
Fig. 2: Hartson & Pyla User Experience Design Lifecycle

RESULTS









DISCUSSION

User Sentiment Analysis (as of July 11, 2024)

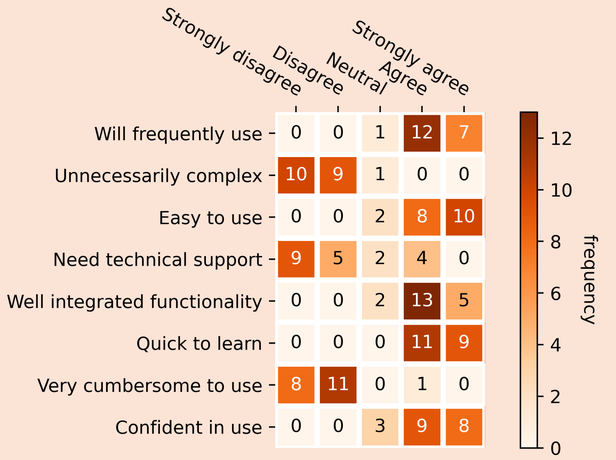


Fig. 3: User-friendly and good tool usability (n=20)

FUTURE WORK

Future work involves reviewing and researching practicality of integrating tool improvements suggested by user study participants, such as an interactive graph with clickable nodes.

ACKNOWLEDGEMENTS

Thank you to Code World: No Blanket, specifically Dr. Chris Brown and Dibyendu Brinto Bose, and the Multicultural Academic Opportunities Program.

User Interface Development using CustomTkinter for Automating Python Dependency Recognition

Travis Chan, Dibyendu Brinto Bose, and Dr. Chris Brown
College of Engineering Department of Computer Science, Virginia Tech, Blacksburg, VA

INTRODUCTION

Timely recognition of and actionability toward resolving Python library dependency issues is crucial in supporting a productive software engineering workflow. Considering not all Python developers have machine learning (ML) knowledge, direct interaction with ML models utilized for backend development is not an ideal user experience. Thus, usability engineering plays a critical role in enabling Python developers with actionable information.

METHODS

A. Understanding needs
- a) Analysis of user work and needs reveals a variety of interaction forms for dependency management tools

B. Designing solutions
- a) "Emotional impact"
- b) "Physical action"
- c) "Existence of feedback"

C. Prototype candidates
- a) User study prototype

D. Evaluate user experience
- a) Nielsen's heuristics
- b) User study to inform tool's impact on software development workflow
  - a. 20 participants (70% M, 30% F)
    - i. 1 to 10+ years of Python experience

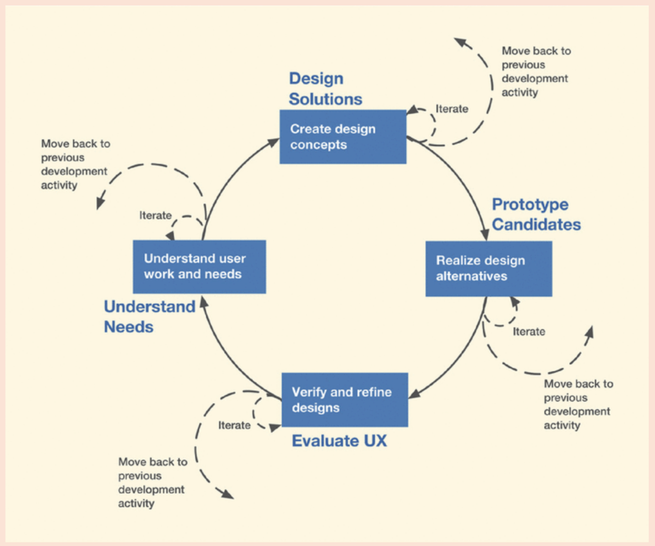
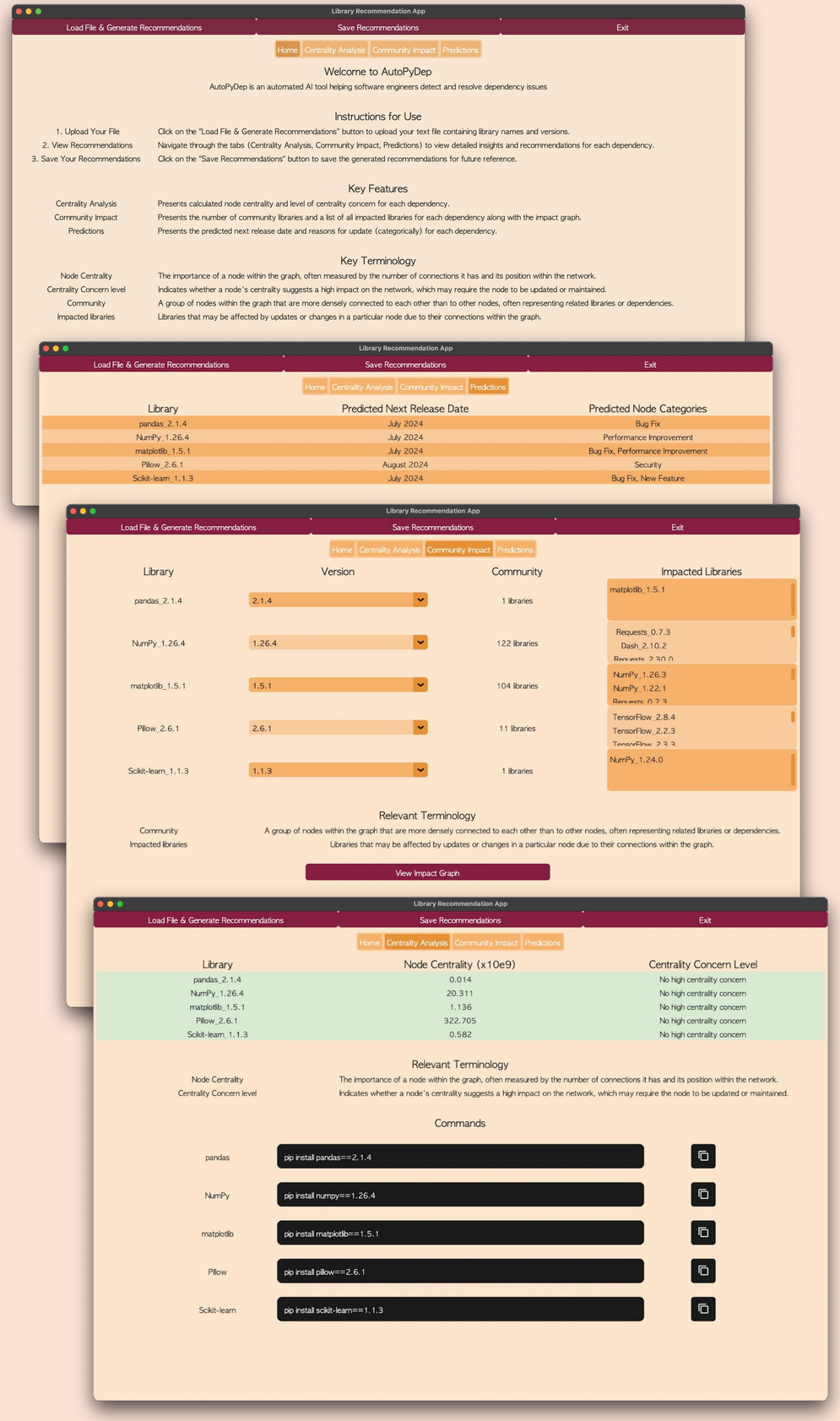
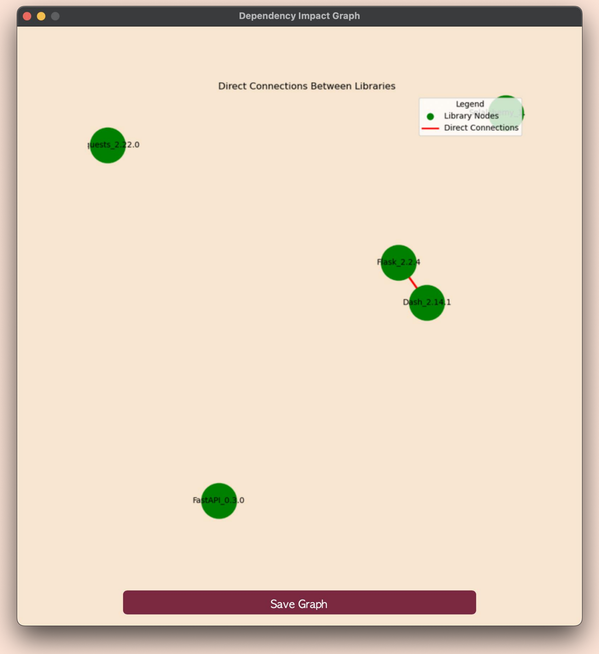


Fig 1. Hartson & Pyla UX Design Lifecycle

RESULTS





DISCUSSION

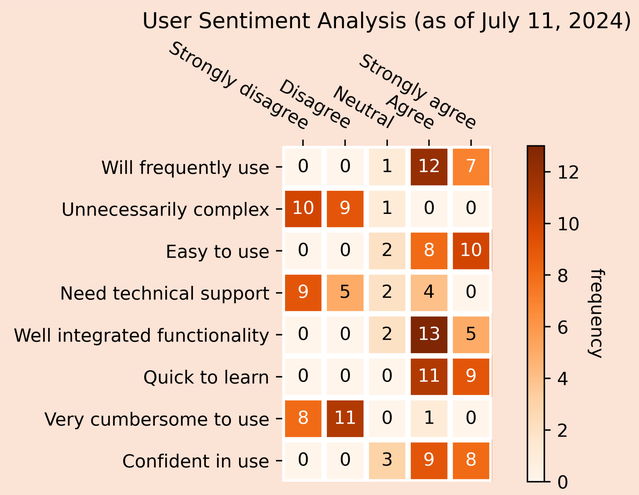


Fig 2. User sentiment analysis heat map

FUTURE WORK

Future work involves reviewing and researching practicality of integrating tool improvements suggested by user study participants.

ACKNOWLEDGEMENTS

Thank you to Code World: No Blanket, specifically Dr. Chris Brown and Dibyendu Brinto Bose, and the Multicultural Academic Opportunities Program.