User interface design

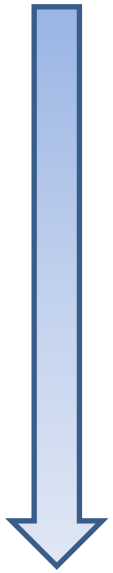
Computing & Information Sciences

W. H. Bell

Software development lifecycle

One iteration

- Requirements definition.
- **Software and systems design.**
- Implementation and unit testing.
- Integration and system testing.
- Operation and maintenance.

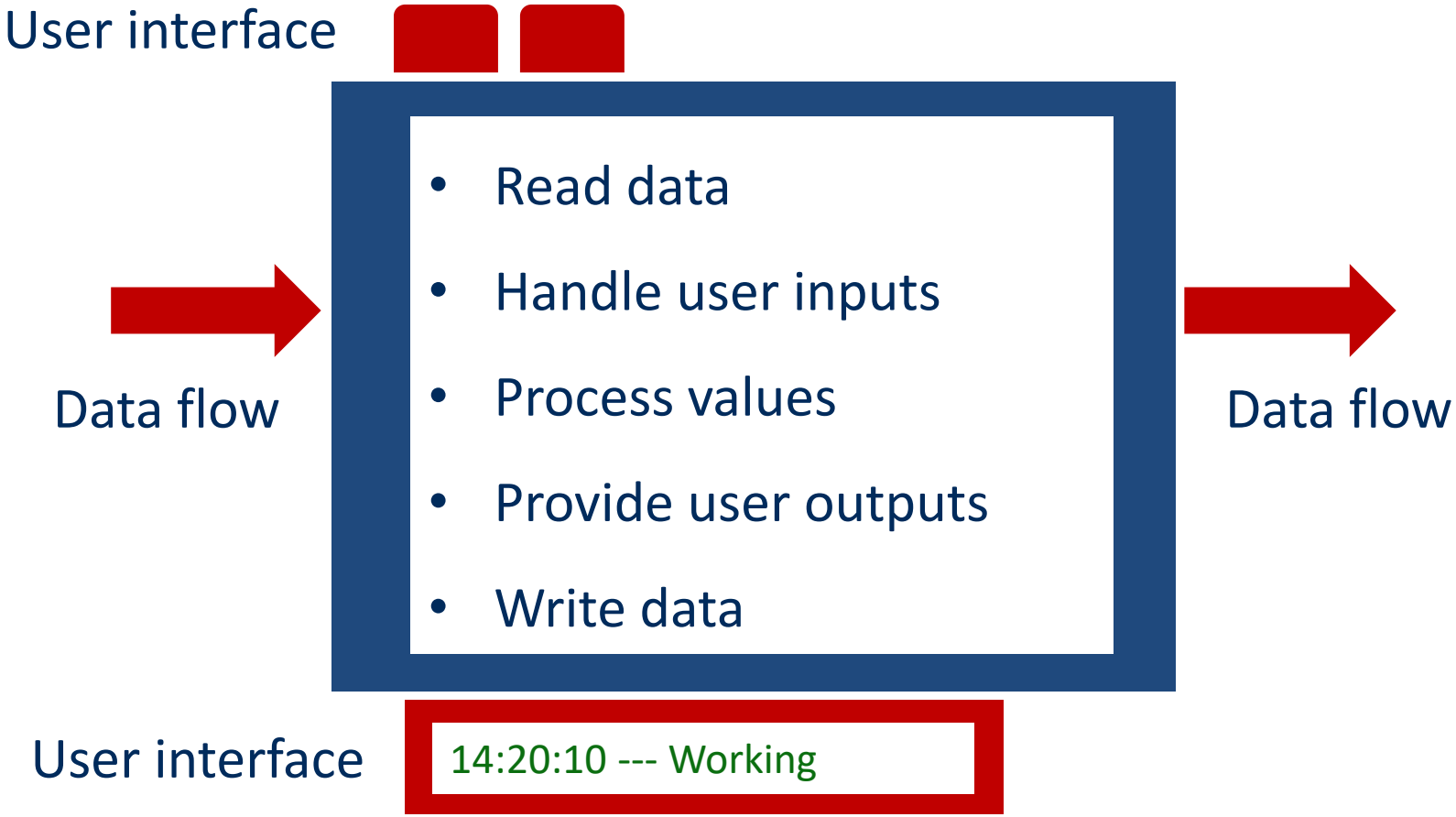


User interface & requirements

University of Strathclyde Glasgow

- Map user interface features to user stories/requirements.
- User interface design enables requirements capture.
  - Users can see more clearly how the software will function.
  - Clear up misunderstandings between developers and users.
  - Additional user stories/requirements should be added.
  - Start to understand data model.

Understand how UI will be used



Types of user interface

University of Strathclyde Glasgow

- Library
  - Use software through function calls.
  - Application programming interface (API) for different languages.
- Command-line interface.
  - Type commands, which are interpreted.

Types of user interface

- Web service – RESTful interface.
  - Use HTTP methods – POST, GET, PUT, DELETE.
- Networked service.
  - Regular protocol, custom protocol.
- Graphical user interface.
  - Mobile device.
  - Desktop environment.

UI design and features

University of Strathclyde Glasgow

- Capture user requirements.
  - Type of user interface – non-functional requirement.
  - User interface features – user stories/functional requirements.
- Need to understand what the users want.
  - Hardware that the interface should run on.
    - Size of screen, colours.
  - How it will be used.
  - Environmental conditions – lighting, accessibility.

Design principles

University of Strathclyde Glasgow

- Logical structure.
  - User should be able to guess operation based on others.
- Similarity to other interfaces.
  - Commands – ctrl-c, ctrl-v.
  - GUI dialogues – Okay, Cancel.
  - Menu systems.
- Common actions should be easier.
  - Reduce total time spent manipulating user interface.

Application programming interfaces

University of Strathclyde Glasgow

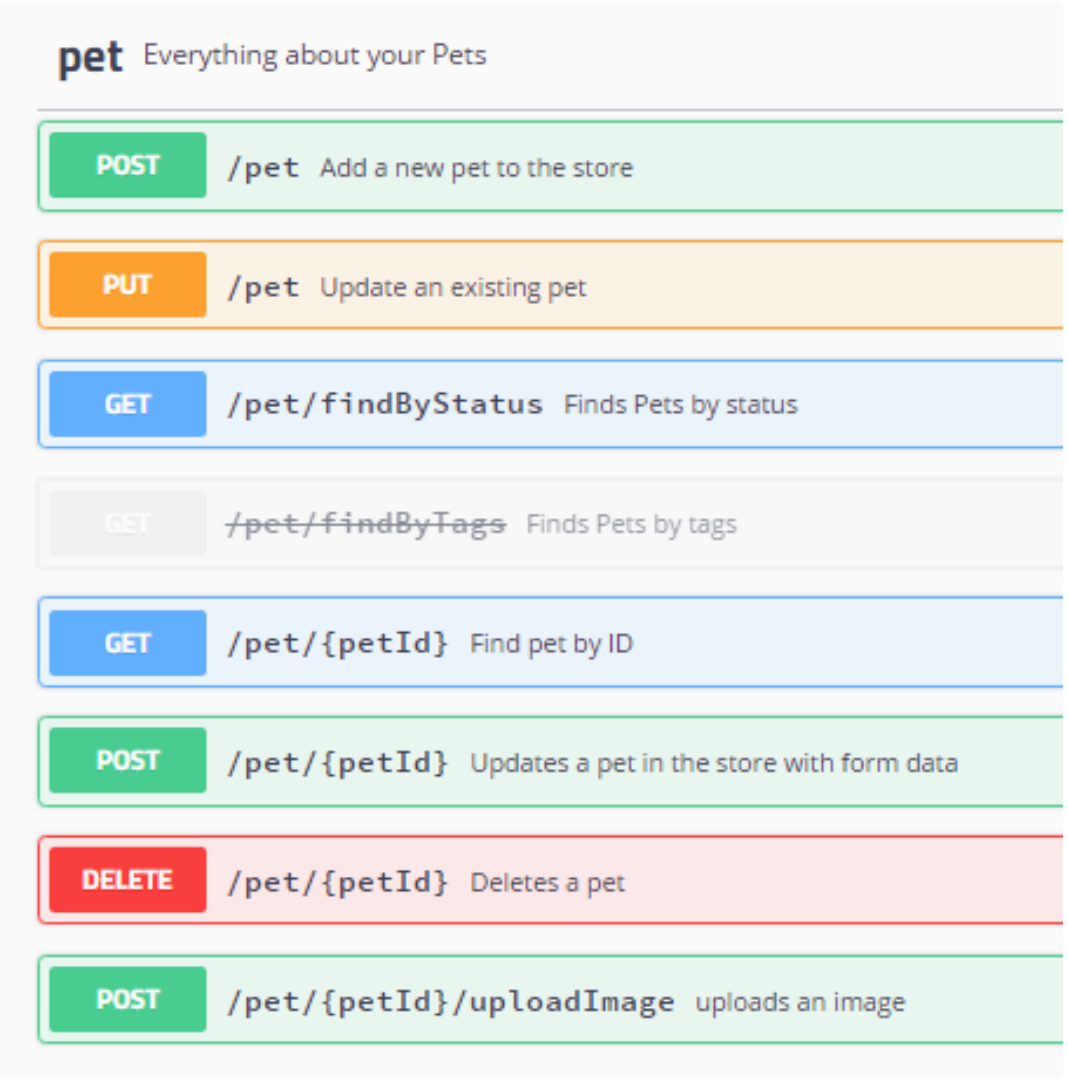
- Design once and then expand.
  - Avoid breaking API changes – affect other software.
  - Use optional fields or more function calls for new features.
  - Backward compatibility needed.
- Logical naming.
  - Function and variable names should be logical.
- Error messages.
  - Helpful information needed to prevent misuse.
- Good documentation.
  - Online, manual pages, built with software, automatically generated.

API design

University of Strathclyde Glasgow

```
STRCMP(3)                  Linux Programmer's Manual                 STRCMP(3)

NAME
       strcmp, strncmp - compare two strings

SYNOPSIS
       #include <string.h>

       int strcmp(const char *s1, const char *s2);

       int strncmp(const char *s1, const char *s2, size_t n);

DESCRIPTION
       The strcmp() function compares the two strings s1 and s2.  The locale
       is not taken into account (for a locale-aware  comparison,  see  str-
       coll(3)).  The comparison is done using unsigned characters.

       strcmp()  returns an integer indicating the result of the comparison,
       as follows:

       • 0, if the s1 and s2 are equal;

 Manual page strcmp(3) line 1 (press h for help or q to quit)
```

Web service – RESTful interface



https://swagger.io/tools/swagger-ui/

Graphical user interface design

University of Strathclyde Glasgow

- Rough sketches.
  - Flipchart or white board discussion.
  - Stakeholder workshop or focus group meeting.
- Initial design using wireframes.
  - No functionality.
  - Talk users through user interface.
  - Refine through workshop or focus groups.
  - Evaluate against other designs and design principles.

Graphical user interface design

University of Strathclyde Glasgow

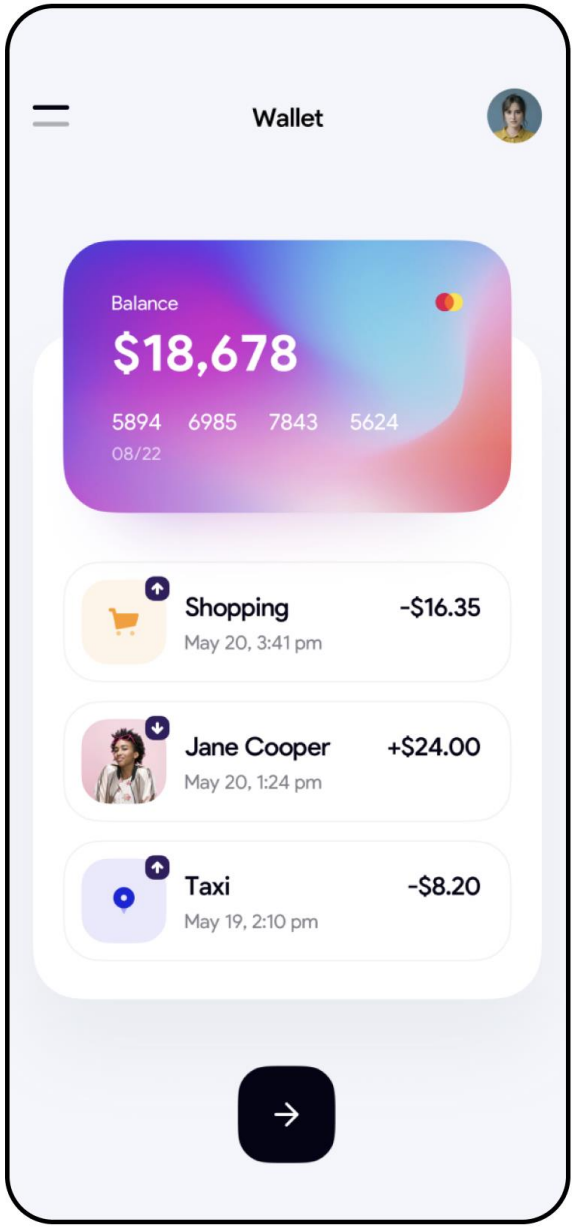
- Wireframes with functionality.
  - Links from one element to another.
  - Use web application or PDF with links.
  - Users can evaluate menu flow.
  - Requires more time to create – cost vs benefit.
  - Allow more users to provide feedback on GUI.

Graphical user interface design

University of Strathclyde Glasgow

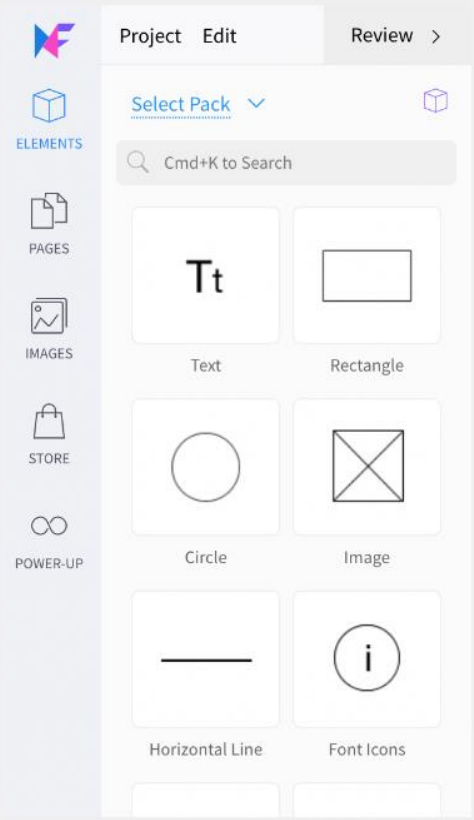
- Initial version of GUI.
  - GUI features without active code.
  - Buttons, textbox, menu, images are placed but inactive.
  - Can include some basic data in UI framework.
- Separate GUI development and discussion.
  - Time needed to move code to new GUI.
  - Implement code once GUI features have been agreed.

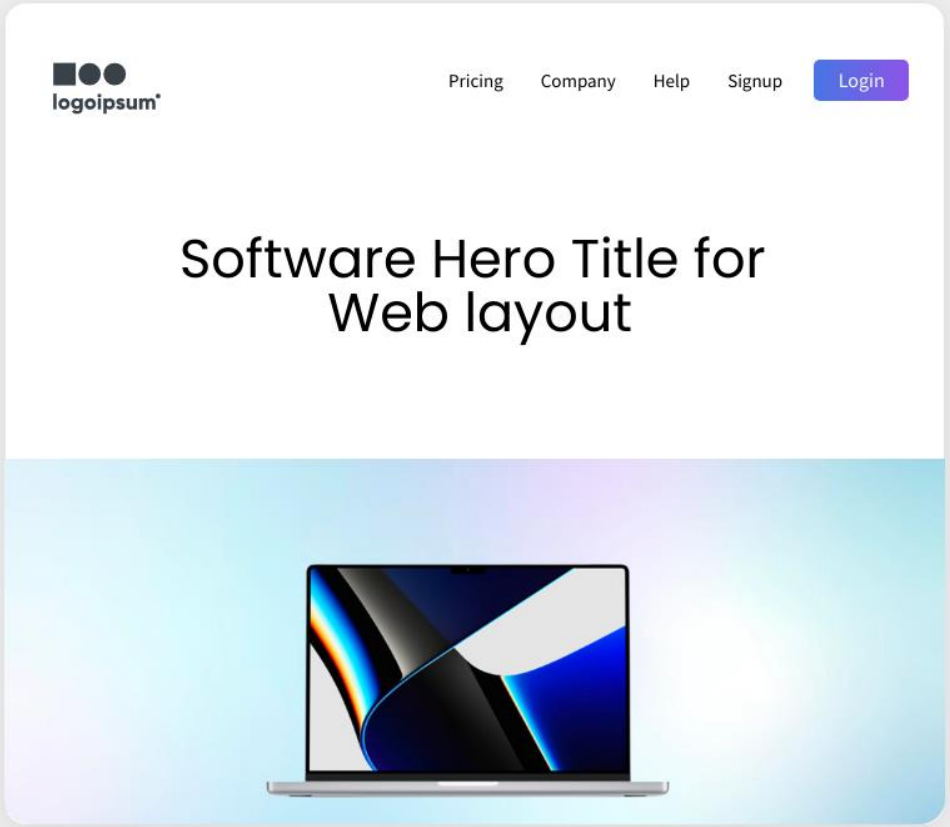
GUI design

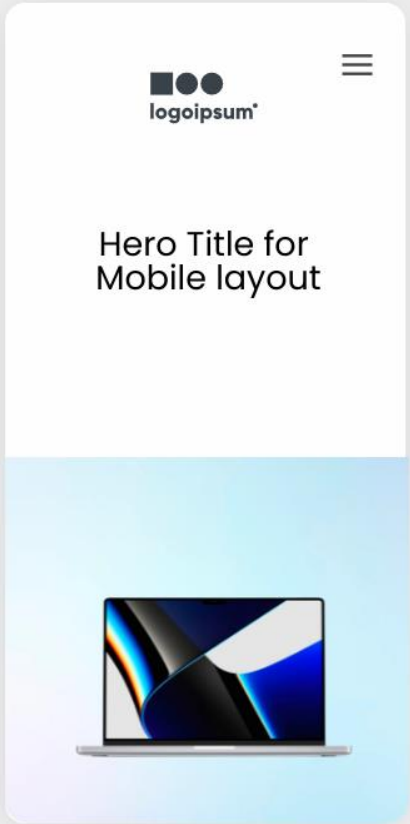


https://www.figma.com/

GUI design







https://www.mockflow.com/

Other GUI design tools

University of Strathclyde Glasgow

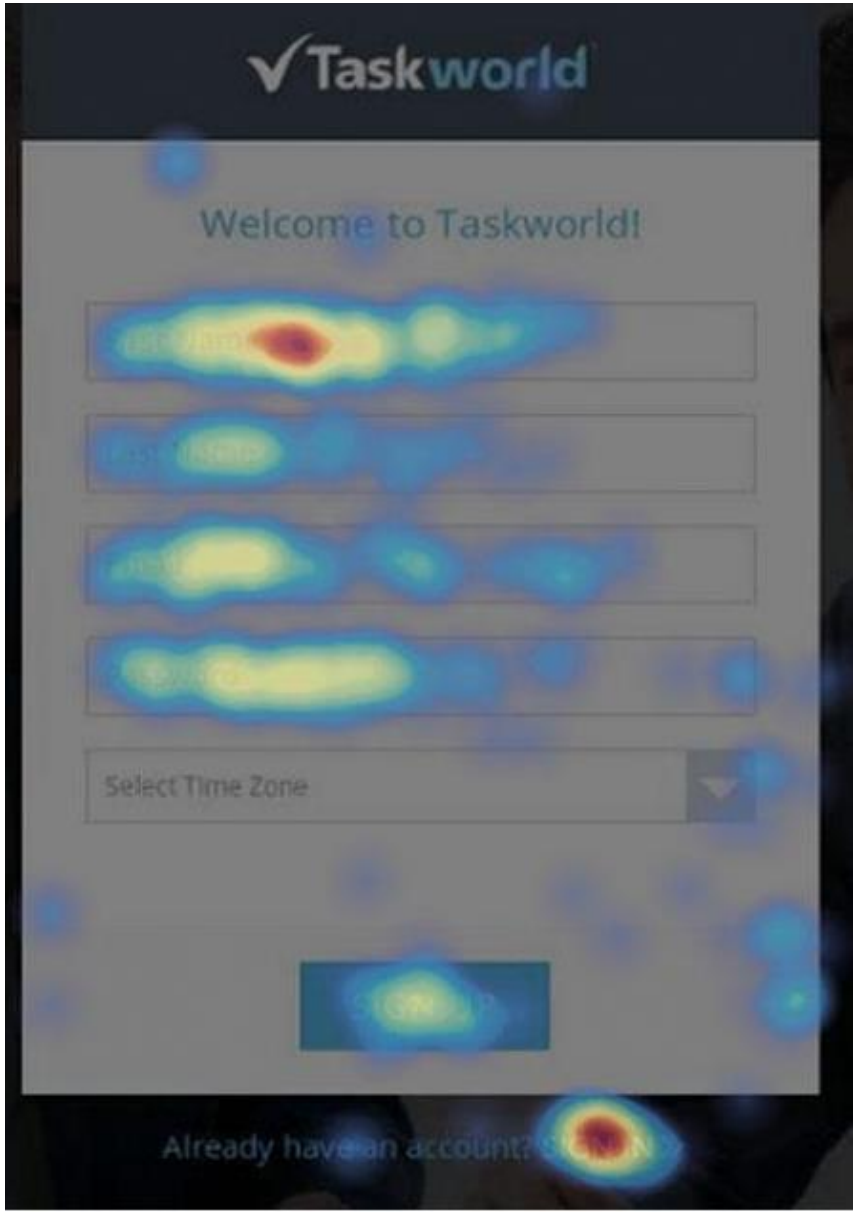
- Adobe XD - https://helpx.adobe.com/uk/support/xd.html
- Balsamiq - https://balsamiq.com/
- Proto.io - https://proto.io/
- Marvel - https://marvelapp.com/
- Sketch - https://www.sketch.com/
- Evolus Pencil - https://pencil.evolus.vn/
- Microsoft Visio

GUI workflows

University of Strathclyde Glasgow

- Sign up – once only.
  - Registration workflow.
- Landing page – every interaction.
  - Customisable dashboard featuring popular data.
- View, edit or save.
  - Reach action quickly.
  - Verify behaviour with analysis, observation.

GUI insights



https://www.hotjar.com/

Colour and shapes

- Vision detects the edges of shapes.
- Colour perception depends on surroundings.
  - Avoid colours that are very similar.
- Display must work for red-green colour blind.
  - Evaluate design in black and white or with filter.

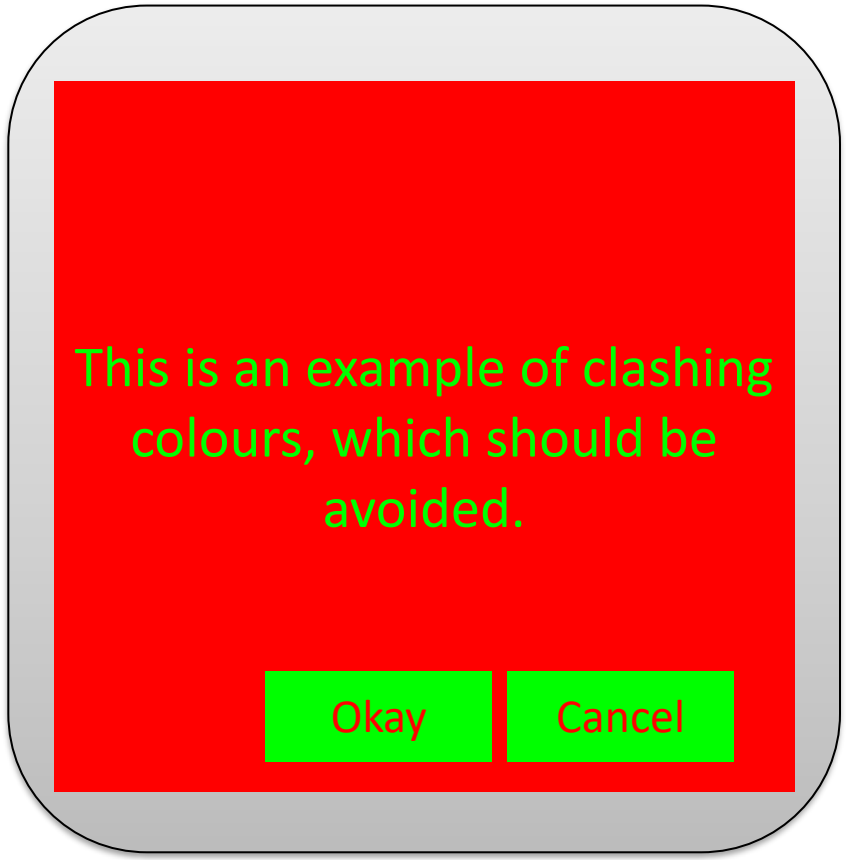
Layout and presentation

- Restrict density of information.
  - Need to include whitespace around design elements.
  - Avoid creating a wall of text.
- Group user interface features.
  - Visually grouped features – assumed to be a group.

Layout and presentation

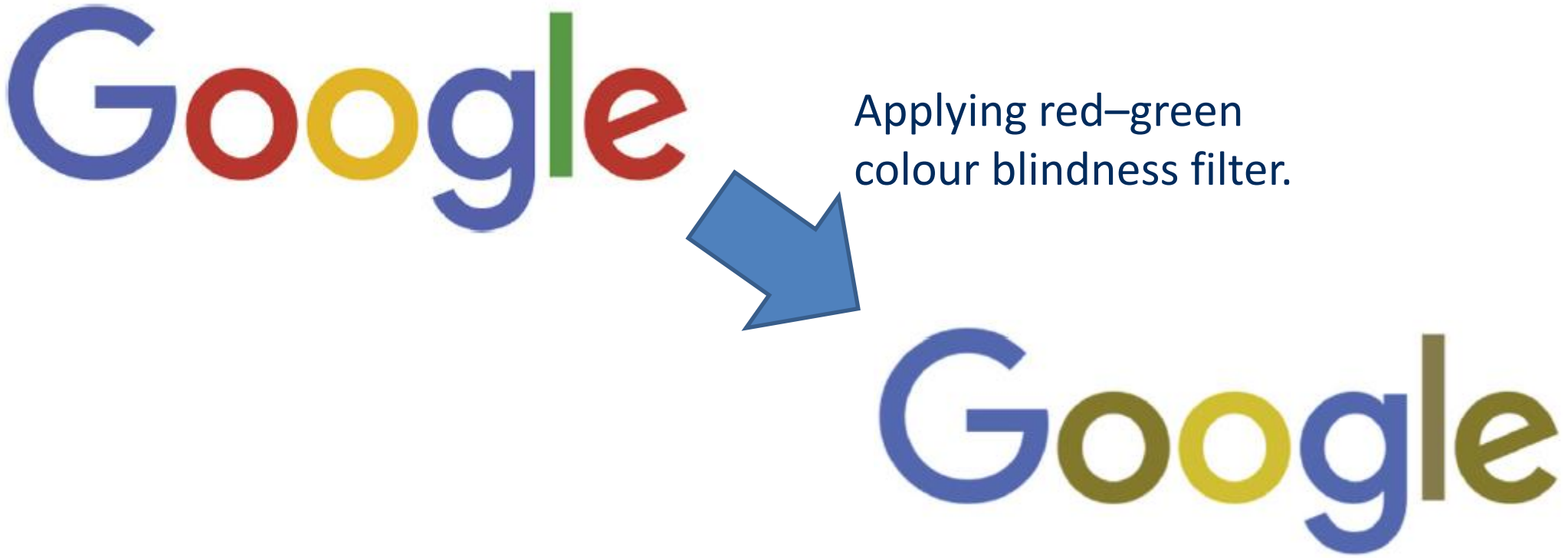
- Use fonts and sizes carefully.
  - Increased line spacing.
  - Dyslexic font option.
  - Implement using library with font accessibility options.

Avoid clashing colours



- Use muted colours.
  - Eye is most sensitive to green.
  - Avoid light green for text.
- Avoid opposite colours.
  - Avoid red and green.
  - Avoid blue and yellow.

Colour blindness



J. Johnson, “Designing with the mind in mind: simple guide to understanding user interface design guidelines”, 2021

User interface evaluation

University of Strathclyde Glasgow

- Evaluation throughout design process.
  - Cannot afford to deliver software with bad UI design.
  - Fix price against UI design.
  - May allow UI design changes in Agile development.

Heuristic evaluation

- Usability analysis.
  - Involve focus group of users.
  - Users evaluate what is good and bad.
  - Evaluate against usability heuristics.

Heuristic evaluation

1. Simple and natural dialogue.
2. Speak the user's language.
3. Minimise user memory load.
4. Be consistent.
5. Provide feedback.
6. Provide clearly marked exits.
7. Provide shortcuts.
8. Good error message.
9. Prevent errors.

J. Nielsen and R. Molich, "Heuristic evaluation of user interfaces", Proceedings of the SIGCHI conference on Human factors in computing systems, 1990.

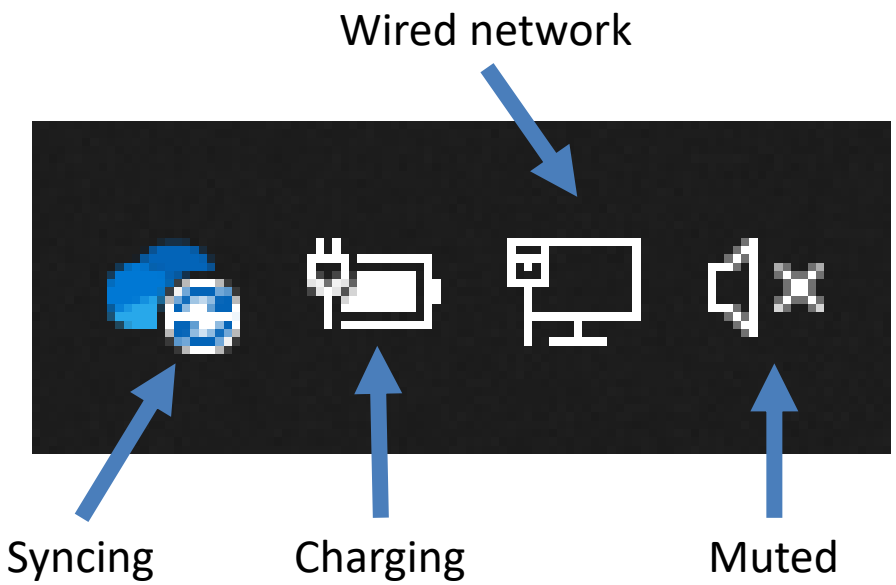
Heuristic evaluation

1. Visibility of system status.
2. Match between system and the real world.
3. User control and freedom.
4. Consistency and standards.
5. Error prevention.
6. Recognition rather than recall.
7. Flexibility and efficiency of use.
8. Aesthetic and minimalist design.
9. Help users recognize, diagnose, and recover from errors.
10. Help and documentation.

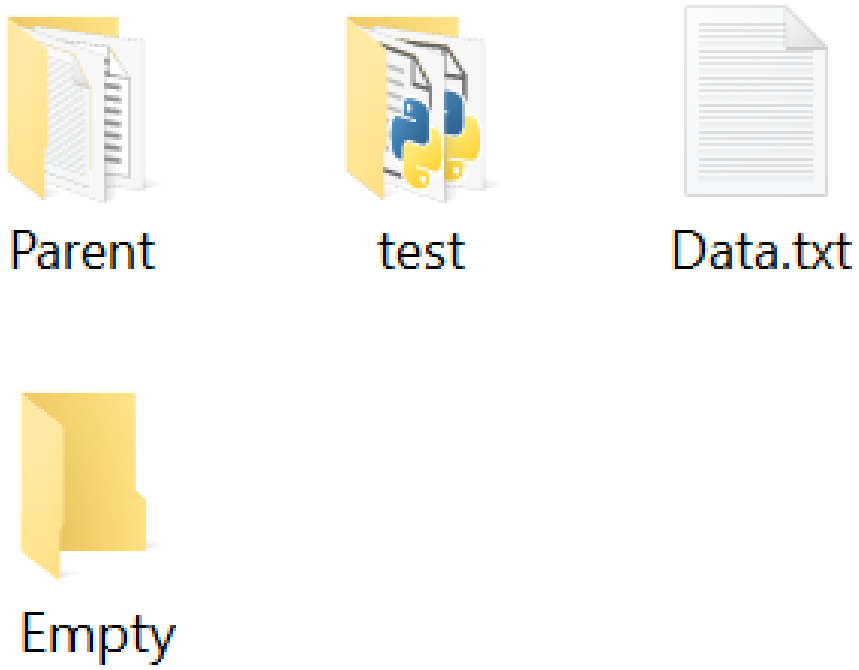
J. Nielsen, “Ten usability heuristics”, 2005

Visibility of system status

- Progress bar in dialogue box.
  - Update to match relative progress.
- Status message at bottom of window or system tray.
  - E.g. “connected”, “disconnected”.
  - Can use system status icons.
- Access system errors.

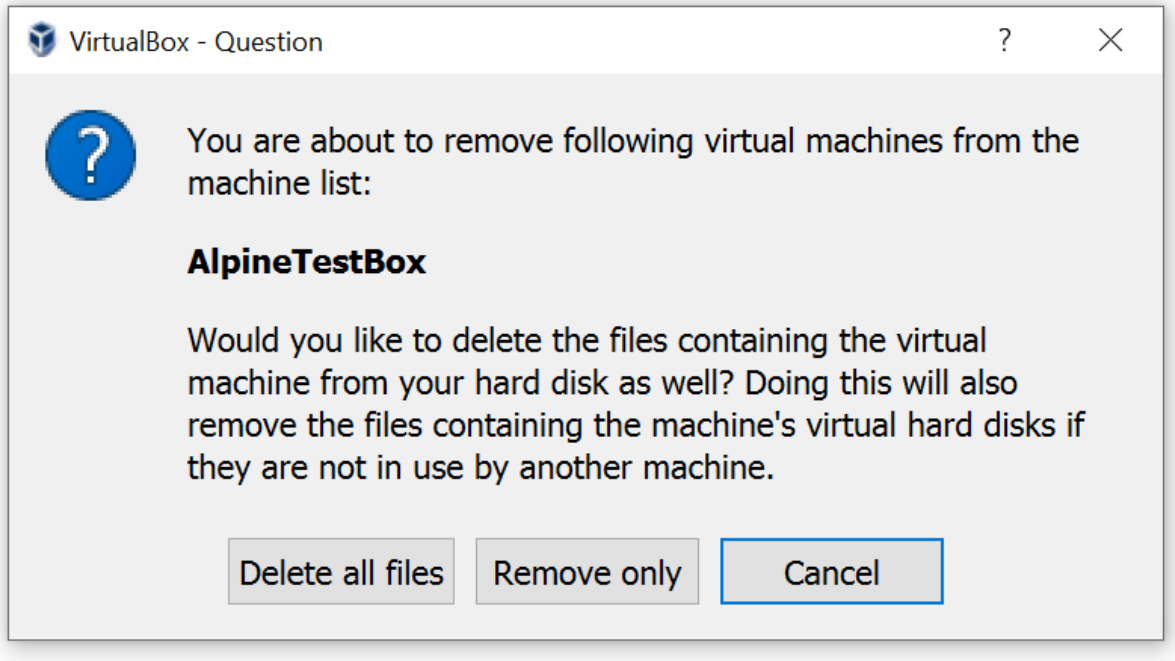


System and real world



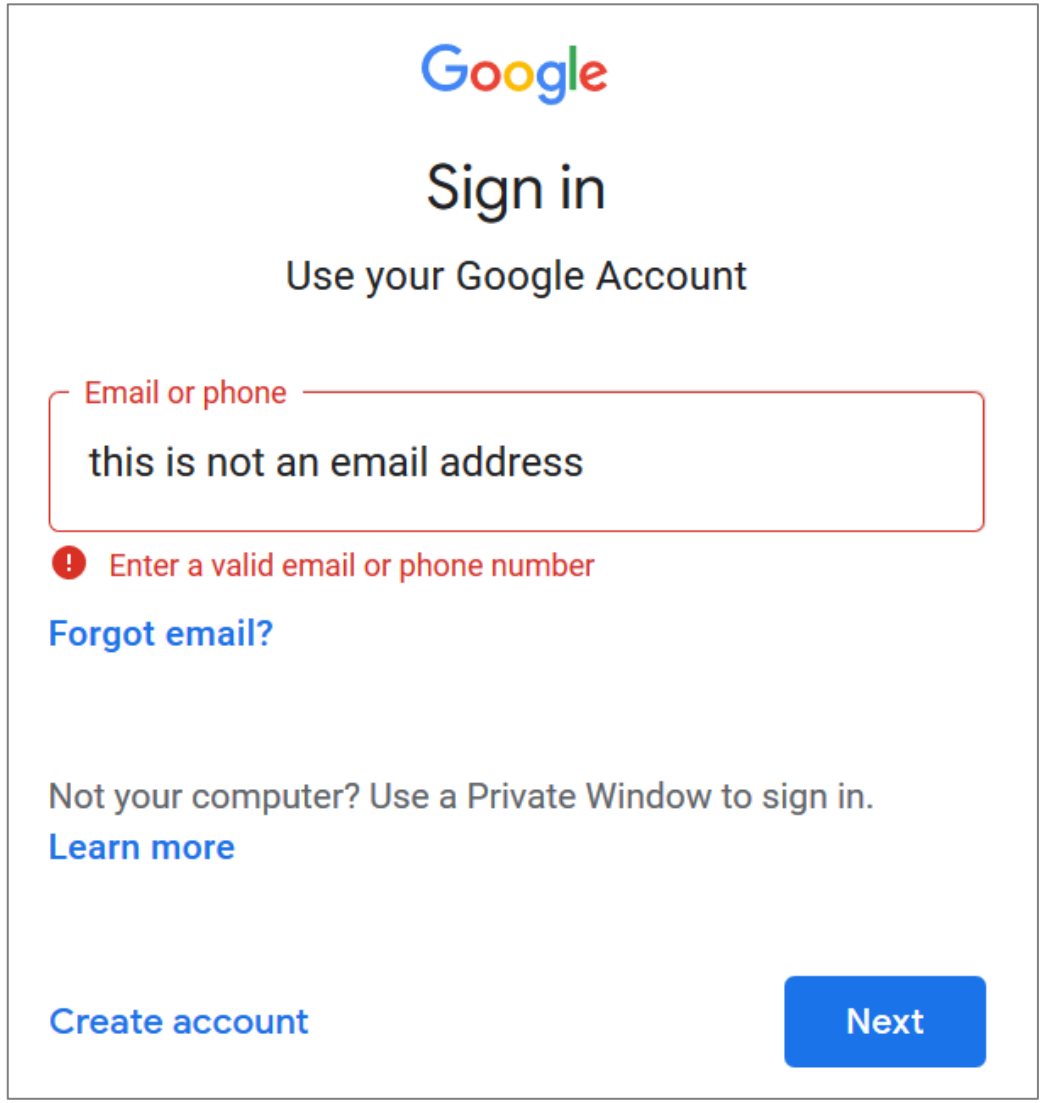
- Files appear as documents.
  - Indicate type of document.
- Folders contain files.
  - Empty folders appear empty.

User control



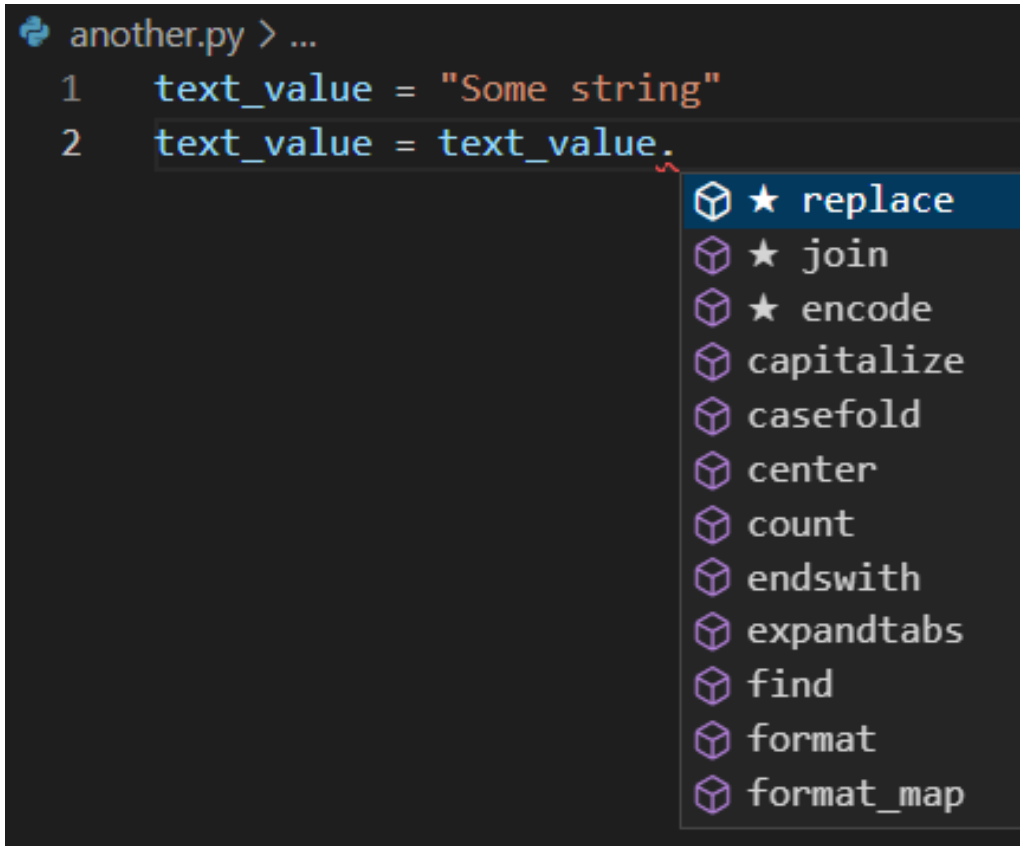
- Provide navigation.
  - Include emergency exits.
- Expect accidental requests.
  - Default to less dangerous outcome.

Error prevention



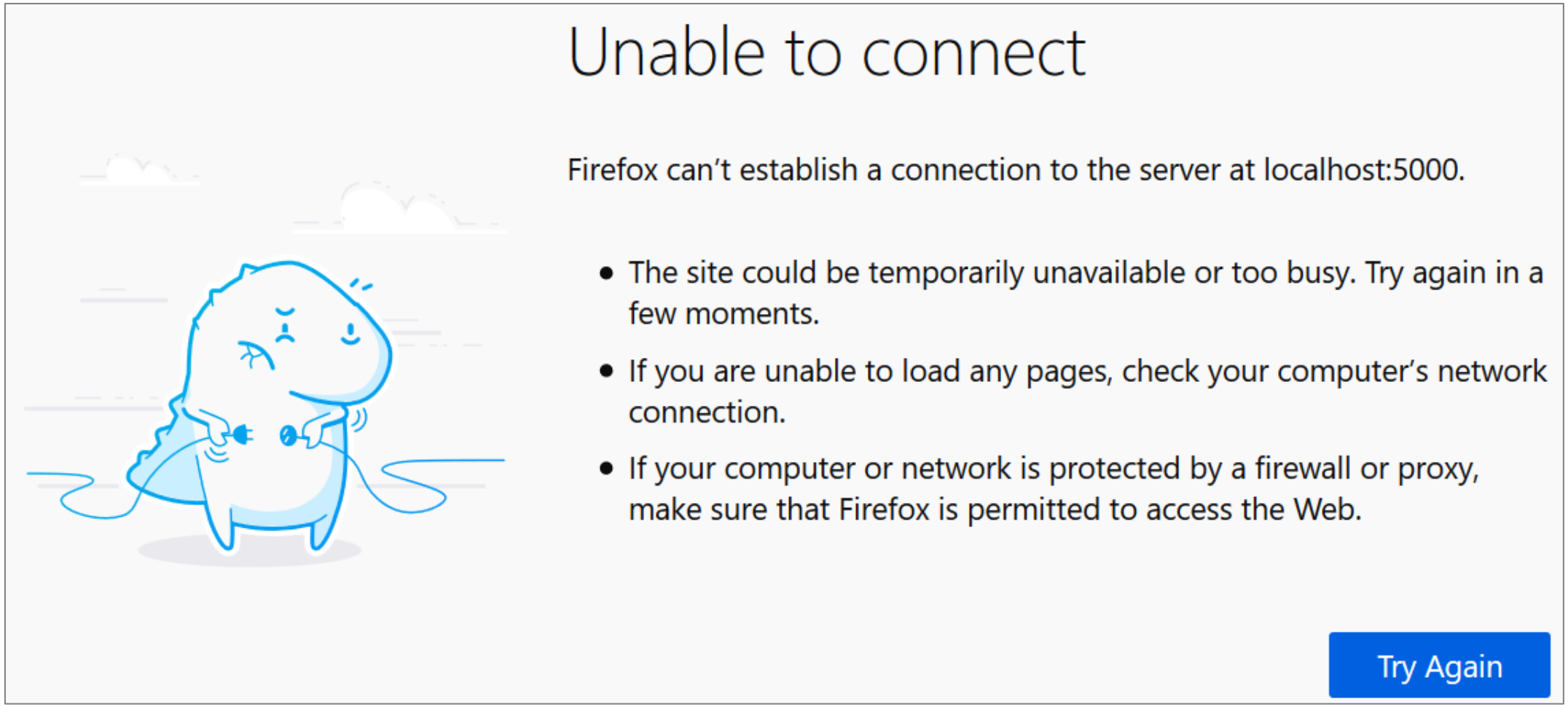
- Required input fields.
  - Do not allow user to proceed.
- Required input types.
  - Text or number.
- Limited input values.
  - Valid numbers (e.g. > 0).
  - Text that has required form.
    - Email.
  - Limited length text.
  - No SQL queries!

Recognition rather than recall



- Dropdown menu.
  - Include available options.
- Autocomplete.
  - Filter dropdown.

Help users to diagnose errors



Firefox web browser, when web server is not accessible.

Conclusions

University of Strathclyde Glasgow

- User interface design enables requirements capture.
  - Discussion with users.
  - Data model requirements.
- Success requires high-quality user interface.
  - More important when competing against similar applications.

University of Strathclyde Glasgow