

IIT Madras BSc Degree

Copyright and terms of use

IIT Madras is the sole owner of the content available in this portal - onlinedegree.iitm.ac.in and the content is copyrighted to IIT Madras.

- Learners may download copyrighted material for their use for the purpose of the online program only.
- Except as otherwise expressly permitted under copyright law, no use other than for the purpose of the online program is permitted.
- No copying, redistribution, retransmission, publication or exploitation, commercial or otherwise of material will be permitted without the express permission of IIT Madras.
- Learner acknowledges that he/she does not acquire any ownership rights by downloading copyrighted material.
- Learners may not modify, publish, transmit, participate in the transfer or sale, create derivative works, or in any way exploit, any of the content, in whole or in part.

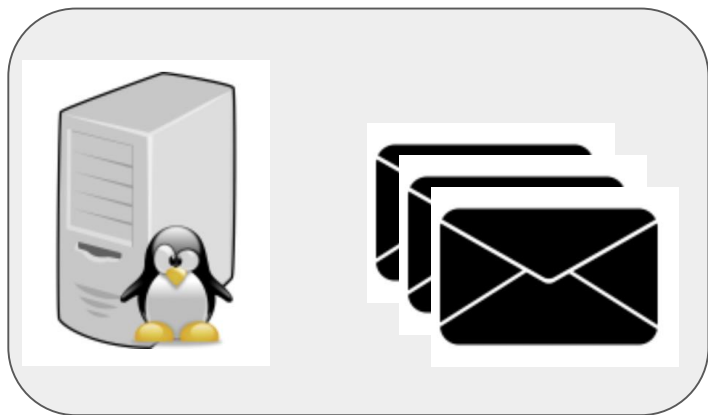
Views

Outline

- MVC paradigm
- Views and User Interfaces
- Tools and Techniques
- Accessibility

MVC


- Model
- **View**
- Controller



Model: Store emails on server,
index, ready to manipulate



View: Display list of emails; Read
individual emails



Controller: Sort emails;
delete; archive

Model-View-Controller

- Origins: Smalltalk-80
- Separation of responsibilities -
Abstraction
- Roots in Object-Oriented GUI development

Model-View-Controller

- Origins: Smalltalk-80
- Separation of responsibilities -
Abstraction
- Roots in Object-Oriented GUI development

Design patterns

- Common software patterns
- **Model**: Application object
- **View**: Screen representation
- **Controller**: How user *interface* reacts to user *input*

Running Example

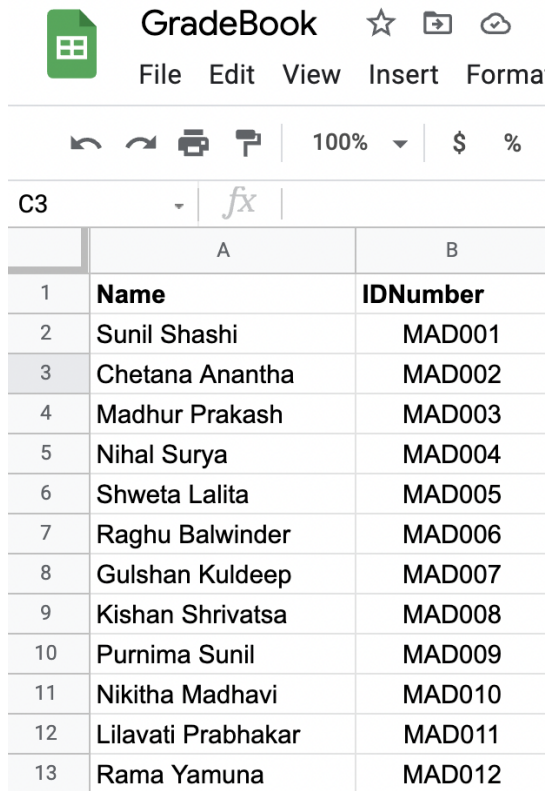
Student Gradebook

- Input data: for **Model**
 - Student list
 - Course list
 - Student-Course marks

Running Example

Student Gradebook

- Input data: for **Model**
 - Student list
 - Course list
 - Student-Course marks



The image shows a screenshot of a web application titled "GradeBook". It features a green spreadsheet icon, a star icon, and a folder icon. Below the title is a menu bar with "File", "Edit", "View", "Insert", and "Forma". A toolbar contains icons for undo, redo, print, and a dropdown menu showing "100%". To the right of the toolbar are currency symbols "\$" and "%". The main area displays a table with columns "A" and "B". The table has 13 rows, with the first row containing headers "Name" and "IDNumber". The subsequent rows list student names and their corresponding IDs.

C3		
	A	B
1	Name	IDNumber
2	Sunil Shashi	MAD001
3	Chetana Anantha	MAD002
4	Madhur Prakash	MAD003
5	Nihal Surya	MAD004
6	Shweta Lalita	MAD005
7	Raghu Balwinder	MAD006
8	Gulshan Kuldeep	MAD007
9	Kishan Shrivatsa	MAD008
10	Purnima Sunil	MAD009
11	Nikitha Madhavi	MAD010
12	Lilavati Prabhakar	MAD011
13	Rama Yamuna	MAD012

Running Example

Student Gradebook

- Input data: for **Model**
 - Student list
 - Course list
 - Student-Course marks

	A	B
1	Name	IDNumber
2	Sunil Shashi	MAD001
3	Chetana Anantha	MAD002
4	Madhur Prakash	MAD003
5	Nihal Surya	MAD004
6	Shweta Lalita	MAD005
7	Raghu Balwinder	MAD006
8	Gulshan Kuldeep	MAD007
9	Kishan Shrivatsa	MAD008
10	Purnima Sunil	MAD009
11	Nikitha Madhavi	MAD010
12	Lilavati Prabhakar	MAD011
13	Rama Yamuna	MAD012

	A	B	C
1	StudentID	CourseID	Marks
2	MAD003	AM1100	31
3	MAD003	ME1100	35
4	MAD001	BT1010	78
5	MAD002	EE1001	30
6	MAD005	EE1001	68
7	MAD009	AM1100	62
8	MAD012	AM1100	77
9	MAD001	BT1010	41
10	MAD007	MA1020	56
11	MAD012	BT1010	52
12	MAD007	ME1100	59
13	MAD009	MA1020	81

Running Example

Student Gradebook

- Outputs: for **Views**

Running Example

Student Gradebook

- Outputs: for **Views**
 - marks for individual student

	A	B	C
1	Sunil Shashi	MAD001	
2			
3	MAD001	BT1010	78
4	MAD001	MA1020	41
5	MAD001	EE1001	43
6	MAD001	AM1100	96

Running Example

Student Gradebook

- Outputs: for **Views**
 - marks for individual student
 - summary for course
 - histograms

	A	B	C
1	Sunil Shashi	MAD001	
2			
3	MAD001	BT1010	78
4	MAD001	MA1020	41
5	MAD001	EE1001	43
6	MAD001	AM1100	96

Running Example

Student Gradebook

- Modifications: for **Controllers**
 - add new students
 - add new courses
 - modify marks in course

Views

User Interface Design

View

User Interface

- Screen
- Audio
- Vibration (haptic)
- Motor (door open/close)

View

User Interface

- Screen
- Audio
- Vibration (haptic)
- Motor (door open/close)

User Interaction

- Keyboard / Mouse
- Touchscreen
- Spoken voice
- Custom buttons

User Interaction

- Determined by hardware constraints
- Different target devices possible
- User-Agent information useful to identify context

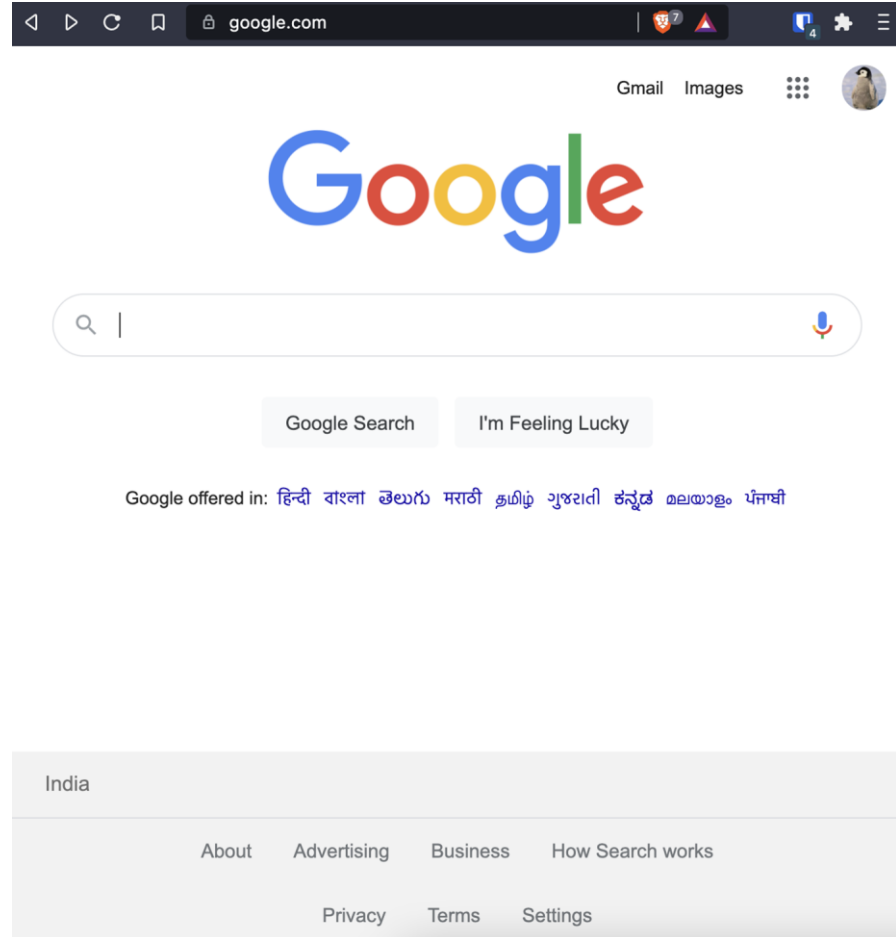
User Interaction

- Determined by hardware constraints
- Different target devices possible
- User-Agent information useful to identify context

May not be under designer control

Types of Views

- Fully static



Types of Views

- Fully static
- Partly dynamic



The screenshot shows the Wikipedia Main Page in English. At the top, the browser address bar displays 'en.wikipedia.org/wiki/Main_Page'. The page features a navigation bar with links for 'Main Page', 'Talk', 'Read', 'View source', and 'View history', along with a search box. Below this, a large banner welcomes users to Wikipedia, stating it is a free encyclopedia that anyone can edit, with 6,339,568 articles in English. The page is divided into several sections: a left sidebar with links to 'Main page', 'Contents', 'Current events', 'Random article', 'About Wikipedia', 'Contact us', 'Donate', 'Contribute', 'Help', 'Learn to edit', 'Community portal', 'Recent changes', 'Upload file', 'Tools', 'What links here', 'Related changes', 'Special pages', 'Permanent link', 'Page information', 'Wikidata item', and 'Print/export'; a central 'From today's featured article' section featuring a photo of Nick Stahl and text about the TV series 'Carnivàle'; and a right sidebar titled 'In the news' with a photo of flooding in Clausen, Luxembourg, and a list of recent news items including 'Floods across western Europe' and 'protests and subsequent unrest' in South Africa.

en.wikipedia.org/wiki/Main_Page

Not logged in Talk Contributions Create account Log in

Main Page Talk Read View source View history Search Wikipedia: Q

Welcome to **Wikipedia**,
the free encyclopedia that anyone can edit.
6,339,568 articles in English

- The arts
- Biography
- Geography
- History
- Mathematics
- Science

From today's featured article

 The two central characters of **Carnivàle**, an HBO television series, were Ben Hawkins (actor pictured), a young man working in a traveling carnival; and Brother Justin Crowe, a Californian preacher. Most of the characters are introduced in Ben's story, though several others interact mainly with Brother Justin; some appear in mysterious dreams and

Nick Stahl, who played Ben Hawkins

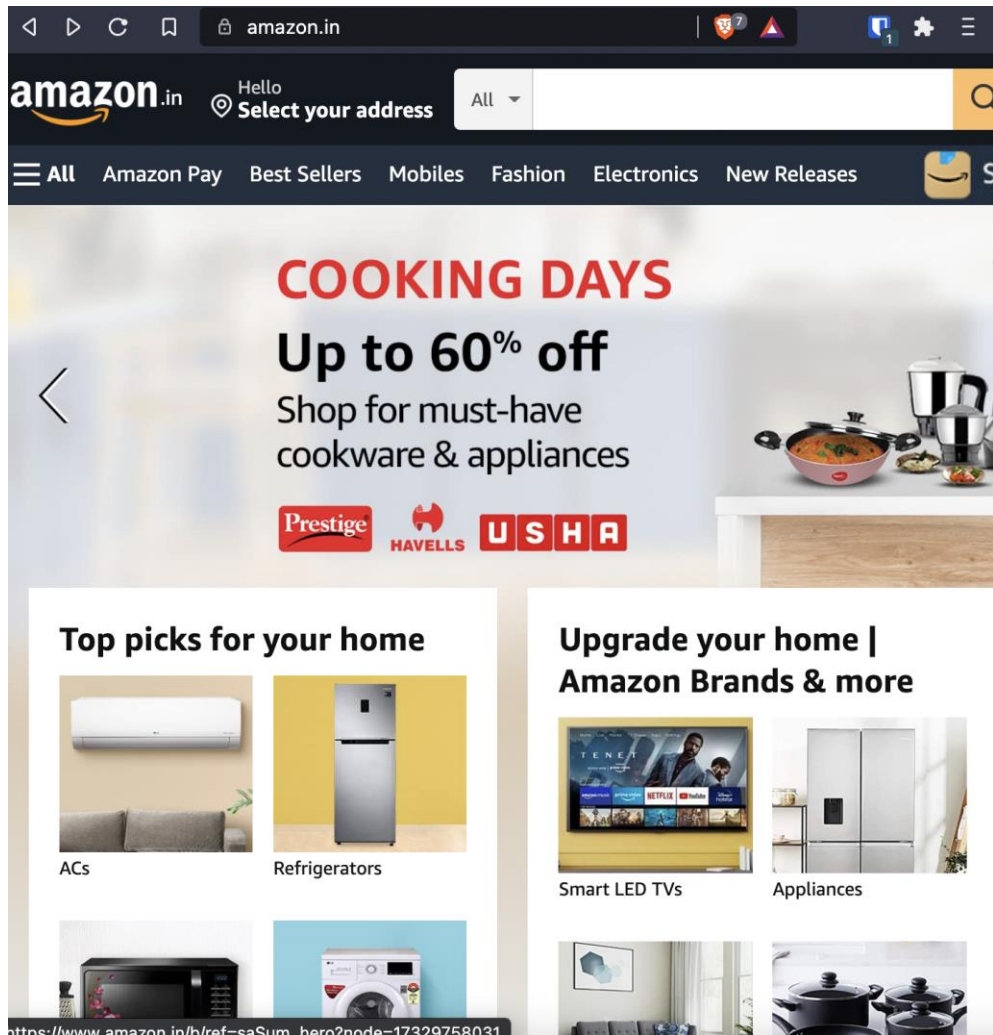
In the news

 Flooding in Clausen, Luxembourg

- **Floods across western Europe** (pictured) leave more than 180 people dead.
- In South Africa, **protests and subsequent unrest** over the jailing of former president Jacob Zuma result in 212 deaths and more than 2,500 arrests.
- The **Supreme Court of**

Types of Views

- Fully static
- Partly dynamic
- Mostly dynamic



Output

- HTML - most commonly used - direct rendering
- Dynamic images
- JSON / XML - machine readable

Output

- HTML - most commonly used - direct rendering
- Dynamic images
- JSON / XML - machine readable

View - any “representation” useful to another entity

User Interface Design

- Design for interaction with user
- Goals:
 - **Simple** - easy for user to understand and use
 - **Efficient** - user achieves goal with minimal effort

User Interface Design

- Design for interaction with user
- Goals:
 - **Simple** - easy for user to understand and use
 - **Efficient** - user achieves goal with minimal effort
- Aesthetics

User Interface Design

- Design for interaction with user
- Goals:
 - **Simple** - easy for user to understand and use
 - **Efficient** - user achieves goal with minimal effort
- Aesthetics
- Accessibility

Systematic Process

- Functionality requirements gathering - what is needed?

Systematic Process

- Functionality requirements gathering - what is needed?
- User and Task analysis - user preferences, task needs

Systematic Process

- Functionality requirements gathering - what is needed?
- User and Task analysis - user preferences, task needs
- Prototyping - wireframes, mockups

Systematic Process

- Functionality requirements gathering - what is needed?
- User and Task analysis - user preferences, task needs
- Prototyping - wireframes, mockups
- Testing - user acceptance, usability, accessibility

Guidelines / Heuristics

Jakob Nielsen's heuristics for design

<https://www.nngroup.com/articles/ten-usability-heuristics/>

- Not specific to web apps, or even software UI design
- Very useful and relevant

General principles

- Consistency
- Simple and minimal steps
- Simple language
- Minimal and aesthetically pleasing

Tools

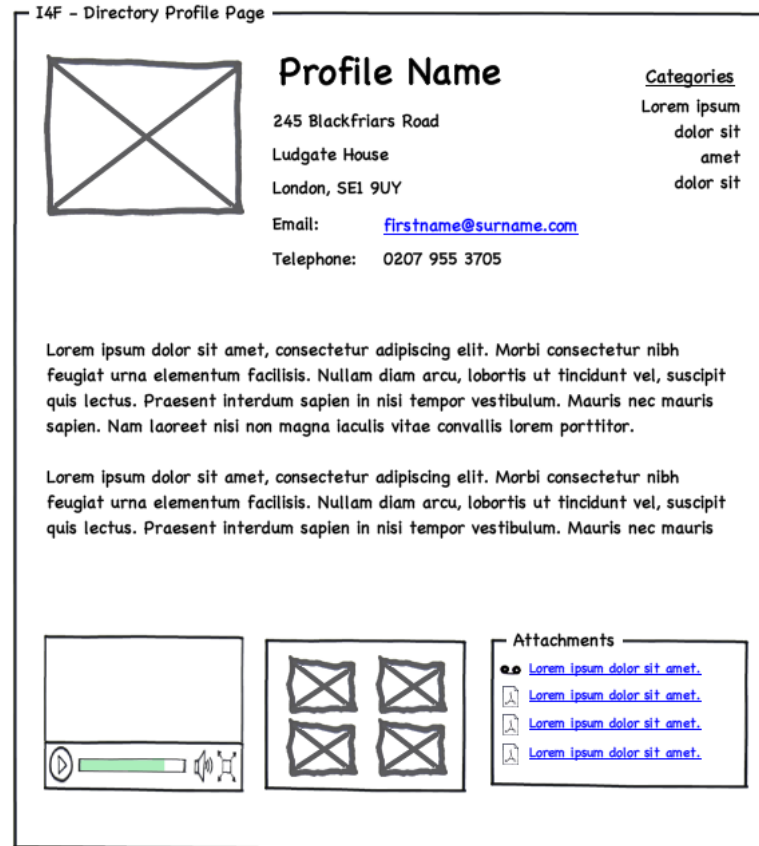
- Wireframes
- HTML generation
- Templates

Wireframes

- Visual guide to represent **structure** of web page
- Information design
- Navigation design
- User interface design

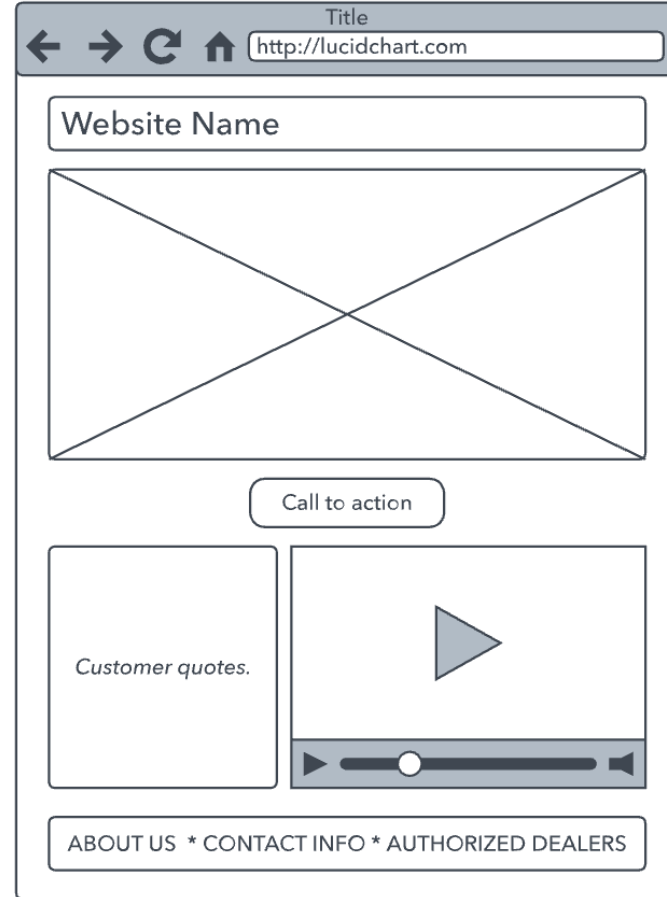
Wireframes

- Visual guide to represent **structure** of web page
- Information design
- Navigation design
- User interface design



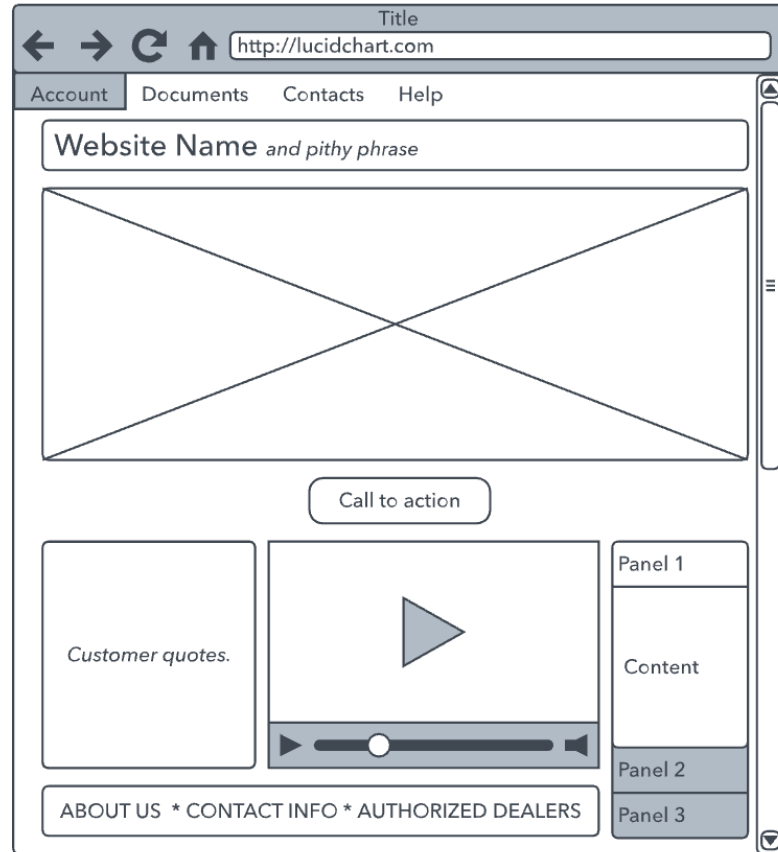
Example Tools: LucidChart

<https://www.lucidchart.com/pages/wireframe>



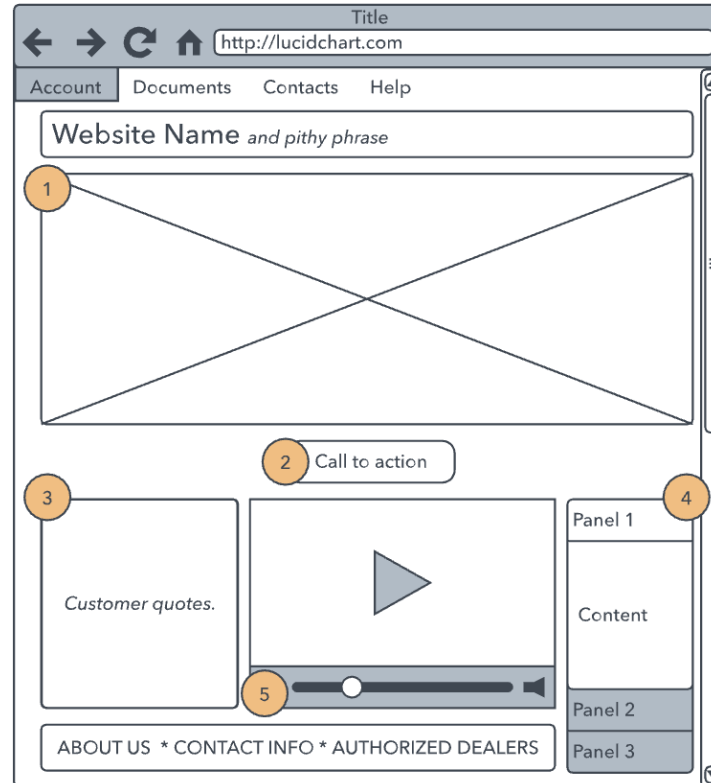
Example Tools: LucidChart

<https://www.lucidchart.com/pages/wireframe>



Example Tools: LucidChart

<https://www.lucidchart.com/pages/wireframe>



Notes

- 1 Details about the image
- 2 Where this links to
- 3 List of possible quotes
- 4 Drop down menus that link to other pages
- 5 Content of video

Programmatic HTML generation: PyHTML

- Composable functions - each function generates a specific output
- Example:
 - to generate a h1 heading: function should return
“<h1>text of heading</h1>”

Programmatic HTML generation: PyHTML

main.py

```
1  import pyhtml as h
2
3  t = h.html(
4      h.head(
5          h.title('Test page')
6      ),
7      h.body(
8          h.h1('This is a title'),
9          h.div('This is some text'),
10         h.div(h.h2('inside title'),
11             h.p('some text in a paragraph.'))
12     )
13 )
14 print(t.render())
```

Programmatic HTML generation: PyHTML

main.py

```
1 import pyhtml as h
2
3 t = h.html(
4     h.head(
5         h.title('Test page')
6     ),
7     h.body(
8         h.h1('This is a title'),
9         h.div('This is some text'),
10        h.div(h.h2('inside title'),
11            h.p('some text in a paragraph.'))
12    )
13 )
14 print(t.render())
```

Console

Shell

```
<!DOCTYPE html>
<html>
  <head>
    <title>
      Test page
    </title>
  </head>
  <body>
    <h1>
      This is a title
    </h1>
    <div>
      This is some text
    </div>
    <div>
      <h2>
        inside title
      </h2>
      <p>
        some text in a paragraph.
      </p>
    </div>
  </body>
</html>
```

More complex HTML

```
def f_table(ctx):  
    return (tr(  
        td(cell) for cell in row  
    ) for row in ctx['table'])
```

Templates

- Standard template text
- Placeholders / Variables
- Basic (very limited) programmability
- Examples:
 - Python inbuilt String Templates - good for simple tasks
 - Jinja2 - used by Flask
 - Genshi
 - Mako
 - ... - *just pick one and go with it*

Jinja

- Ties in closely with Flask
- Template functionality with detailed API

Remember: templates can generate any output, not just HTML

Accessibility

Accessibility

- Various forms of disability or impairment
 - Vision
 - Speech
 - Touch
 - Sensor-Motor
- Can a page be accessed by people with impairments?
- How can the accessibility of a page be improved?

W3.org - World Wide Web Consortium (W3C) - accessibility guidelines

<https://www.w3.org/WAI/fundamentals/accessibility-principles/>

Standards

Interplay between many components of a page:

- Web content: HTML, images, scripts etc.
- User-agents: desktop browser, mobile browser, speech-oriented browser, assistive devices
- Authoring tools: text editor, word processor, compiler

Principle - Perceivable

- Provide text alternatives for non-text content.
- Provide captions and other alternatives for multimedia.
- Create content that can be presented in different ways, including by assistive technologies, without losing meaning.
- Make it easier for users to see and hear content.

Principle - Operable

- Make all functionality available from a keyboard.
- Give users enough time to read and use content.
- Do not use content that causes seizures or physical reactions.
- Help users navigate and find content.
- Make it easier to use inputs other than keyboard.

Principle - Understandable

- Make text readable and understandable.
- Make content appear and operate in predictable ways.
- Help users avoid and correct mistakes.

Principle - Robust

- Maximize compatibility with current and future user tools.

Techniques

[Contents](#)[Intro](#)[Next Technique: ARIA2](#)

Using the aria-describedby property to provide a descriptive label for user interface controls

Important Information about Techniques

See [Understanding Techniques for WCAG Success Criteria](#) for important information about the usage of these informative techniques and how they relate to the normative WCAG 2.1 success criteria. The Applicability section explains the scope of the technique, and the presence of techniques for a specific technology does not imply that the technology can be used in all situations to create content that meets WCAG 2.1.

Applicability

Technologies that support [Accessible Rich Internet Applications \(WAI-ARIA\)](#).

On this page:

[Important Information about Techniques](#)[Applicability](#)[Description](#)[Examples](#)[Related Resources](#)[Related Techniques](#)[Tests](#)

Aesthetics

- Visual appearance
- VERY important
- Simplicity preferred

Can vary with time!



From: The Verge: iOS: A Visual History

Summary

- View - any output seen by human or machine
- User-interface and User-interaction guidelines
- Accessibility is a core concept!
- Tools for automatic generation, consistent layout