

# **COMP350 - Software Design**

Learning by building software

# I notice ...

# I wonder ...

**Sender Name**

16 July 2024

Recipient Name  
123 New Street  
Town, County, Postcode

Dear Recipient Name,

To get started, just tap or click this placeholder text and begin typing. You can view and edit this document on your Mac, iPad, iPhone, or on iCloud.com.<sup>1</sup>

It's easy to edit text, change fonts and add beautiful graphics. Use paragraph styles to get a consistent look throughout your document. For example, this paragraph uses Body style. You can change it in the Text tab of the Format controls.

To add photos, movies, audio and other objects, tap or click one of the buttons in the toolbar or drag and drop the objects onto the page.

Yours sincerely,

Sender Name

**97** words

I notice ...

I wonder ...

Project.band - Tracks

001.1 BAR 120 BEAT 4/4 Cmaj

Library All Sounds ♦

String Ensemble

Wedding Organ

Search Sounds

Brass > Baroque Organ

Choir ⌄ > Bass Organ

Harp ⌄ > Cathedral Organ

Percussion ⌄ >

Pipe Organ > Deep Organ

Flute Organ

Fugue Organ

Mellow Organ

Noble Organ

Romantic Organ

Toccata Organ

Wedding Organ

Orchestral >

Revert Delete Save...

Track Master Compare Controls EQ

Keyboard Sensitivity

Less Neutral More

Drag the slider to adjust the velocity level of notes you play.

Plug-ins >

Low High Cutoff Ambience Reverb Bass

I notice ...

I wonder ...

The screenshot shows a digital note-taking application interface. At the top, there's a navigation bar with icons for back, forward, search, and other functions. Below the navigation is a breadcrumb trail: Home > Talks > IITGN - Programming and music. To the right of the breadcrumb is a search bar and a set of global icons for user, search, and more. The main content area has a title 'IITGN - Programming and music'. Below the title is a bulleted list:

- **Title:** Computing music: Crossing the left-right brain divide
- **Abstract:** The history of computer music is nearly as old as computing itself. It has therefore borrowed and tracked various paradigms of programming and systems engineering in the course of its development and also introduced niche paradigms of its own. The domain offers a unique way to engage with computing for those with some ties to music, and vice versa. Teaching a class with students having diverse capabilities and interests therefore poses interesting challenges of individual development and group work. For context, I'll present some of the history of the domain and the modes of thinking that the field has employed, with a sample of various "languages", systems and methods in use. I'll then bring up some challenges of teaching it to a class, primarily around how to get students comfortable crossing the left-right brain divide.
- **Ref**
  - Vercoe's CSound
  - Andrew Sorenson's live coding
  - Thor Magnusson's Ixilang
  - Max/MSP and Pure Data
  - SuperCollider
  - Sonic Pi
  - Web Audio API and Javascript
    - <https://gibber.cc/>
  - Joe Armstrong's experiments with Erlang and OSC protocol
  - Synthesis and control
    - MIDI / OSC
    - Wavetable synthesis

I notice ...

I wonder ...

The screenshot shows a Jupyter Notebook interface with two tabs open: 'Untitled.ipynb' and 'Untitled1.ipynb'. The 'Untitled1.ipynb' tab is active and displays the following code:

```
[79]: onehot(i,N) = [if i == j 1.0f0 else 0.0f0 end for j in 1:N]
[79]: onehot (generic function with 1 method)

[80]: function exp_a(M; dt=0.01f0)
    N = size(M)[1]
    eM = typeof(M)(Diagonal(ones(Float32,N)))
    #println(typeof(eM))
    for t in dt:dt:1.0f0
        Mv = eM
        for k in 1:4
            Mv = (1.0f0/Float32(k)) * dt * M * Mv
            eM += Mv
        end
    end
    return eM
end

[80]: exp_a (generic function with 1 method)

[81]: mM = MtlArray(M)
[81]: 1024×1024 MtlMatrix{Float32, Private}:
  0.225695  0.218631   ... -0.378506  0.349709  0.449316
  -0.113412 -0.483146   ... -0.490179 -0.335411 -0.389965
  -0.117555  0.253987   ... -0.32382  0.00401694  0.262235
  -0.339267 -0.251612   ... 0.19488  0.228386 -0.154417
  -0.139629 -0.00747603   ... 0.374516 -0.00837421 -0.363738
  -0.0257248  0.425742   ... -0.396415  0.314875 -0.448106
  -0.272622 -0.390271   ... -0.442614  0.277375 -0.0912812
  0.144097  0.208181   ... -0.355407  0.165621  0.437742
  -0.303445 -0.273298   ... -0.464546 -0.0319704  0.488699
  -0.323551  0.153756   ... -0.036872  0.371999 -0.175321
  -0.0178099 -0.42845   ... 0.196424  0.429441  0.431625
  0.457064  0.421712   ... -0.472138 -0.173739 -0.449819
  -0.402644  0.304837   ... -0.411577  0.453018  0.191801
  ...
  -0.363404  0.23864   ... -0.114432  0.441612  0.16551
```

The notebook is running on 'Julia 1.10.4' and is currently idle. The status bar at the bottom shows 'Mode: Command' and 'Ln 1, Col 1'.



I notice ...

A screenshot of a Google search bar showing the letter 'a' typed in. Below the search bar is a list of search suggestions. Each suggestion includes a small icon to its left and the suggested term in bold. The suggestions are: Amazon, apspdcl, apsche, Anant Ambani (with a small profile picture), Internet personality, APSRTC (with a small logo), age calculator, Andhra Pradesh (with a small map icon), State of India, and ap eamcet.

-  **Amazon**
-  **apsdcl**
-  **apsche**
-  **Anant Ambani**  
Internet personality
-  **APSRTC**
-  **age calculator**
-  **Andhra Pradesh**  
State of India
-  **ap eamcet**

I wonder ...

# What does it take?

- Ideas
- Grounding and validation
- Planning and detailing of behaviour
- **Architecture:** Systems and components involved
- **Design:** Appearance and interactions
- **Code Design:** functions / modules / packages, their **interfaces** and **contracts**
- Coding, testing, debugging

# What does it take? - Ethics

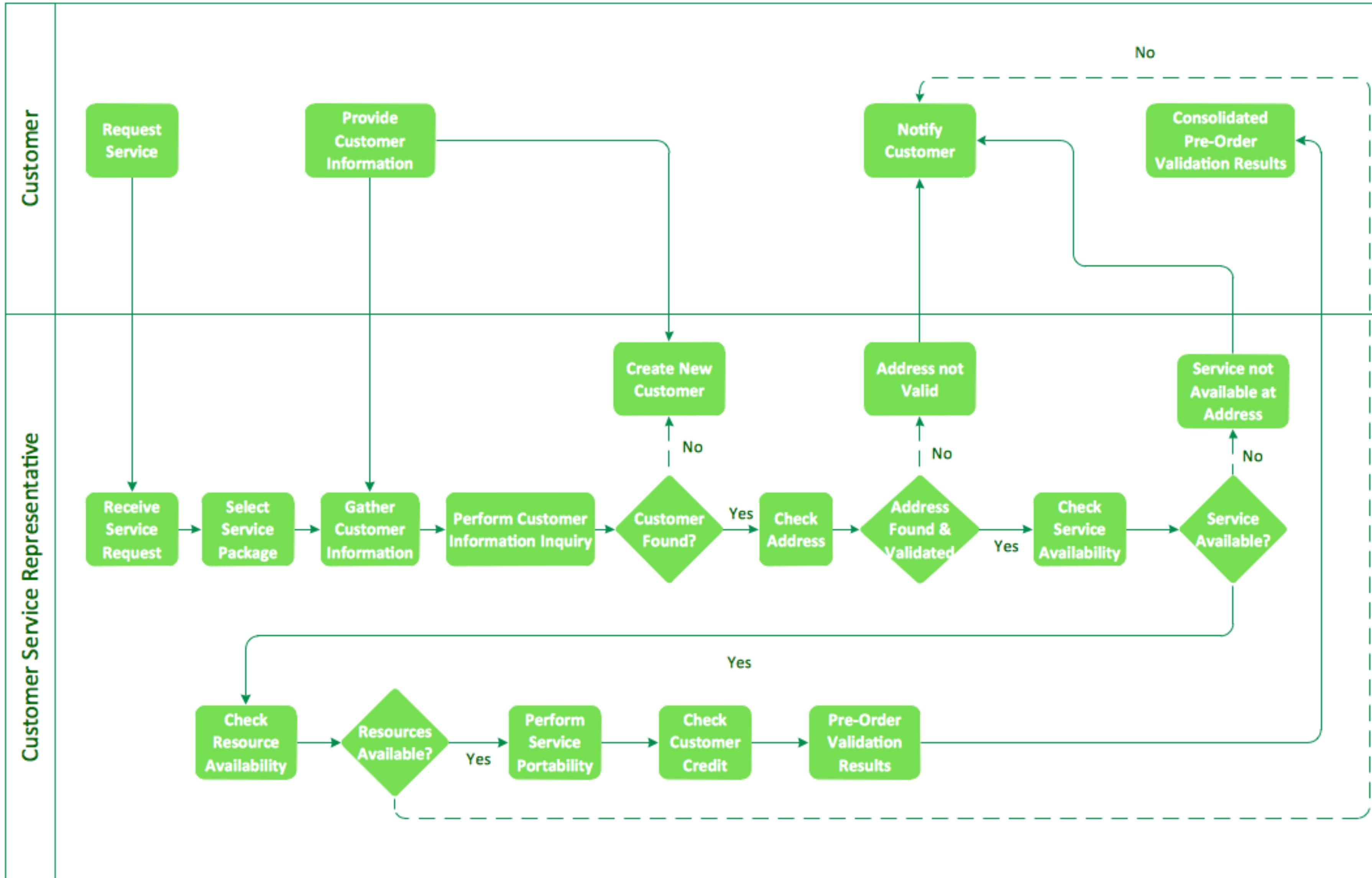
- Accessibility considerations
  - Readability, Colour blindness, Blindness, Physical disabilities, ...
- Algorithmic biases and responsible design
- Correctness and consequences
- Team affordances and accommodations

# Some questions to keep in mind

- **Who** is going to use your software?
- **What** are they doing with it?
- **How** are they going to go about doing it?
- **How** do we plan to find out and improve it?

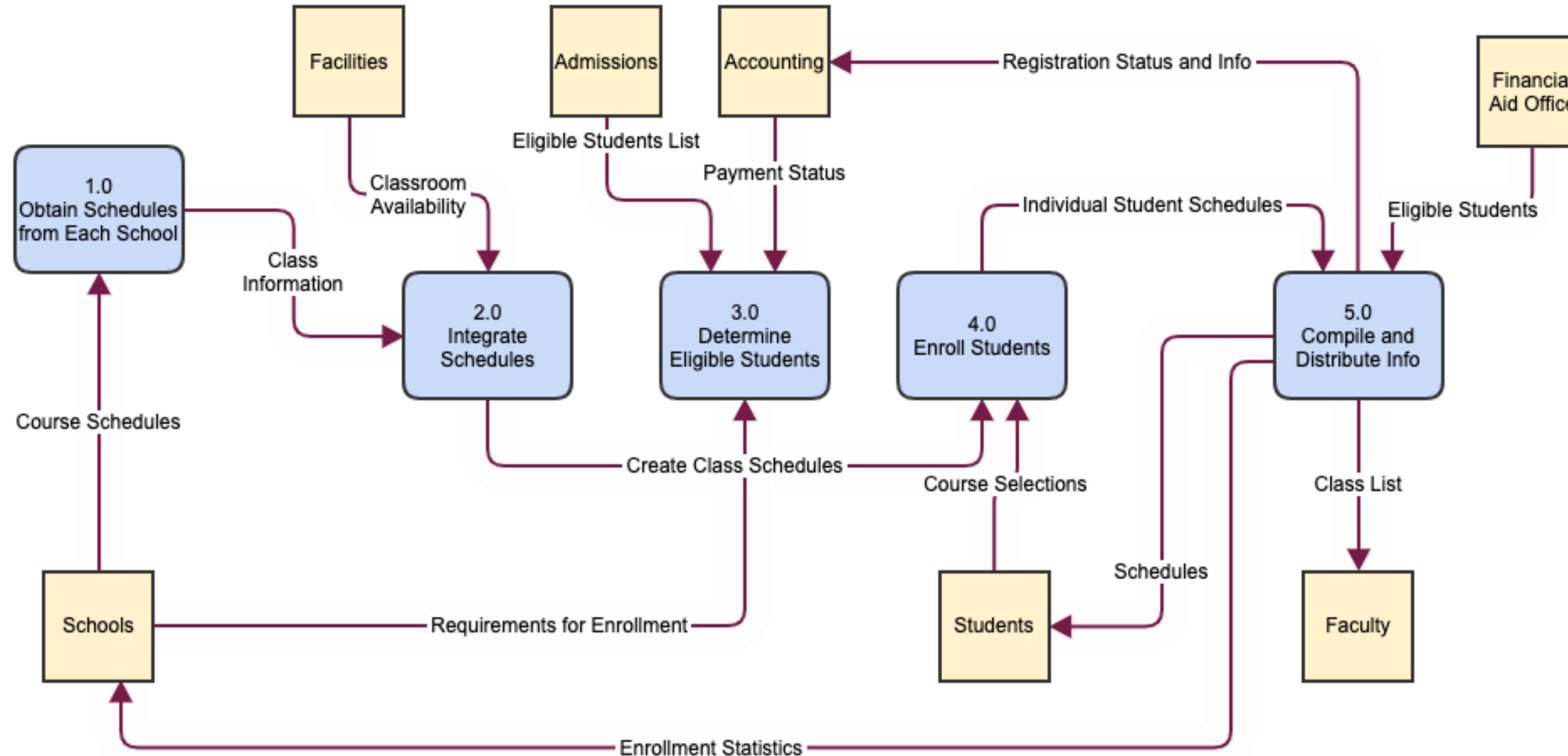
# Artifacts

# Process flow diagram



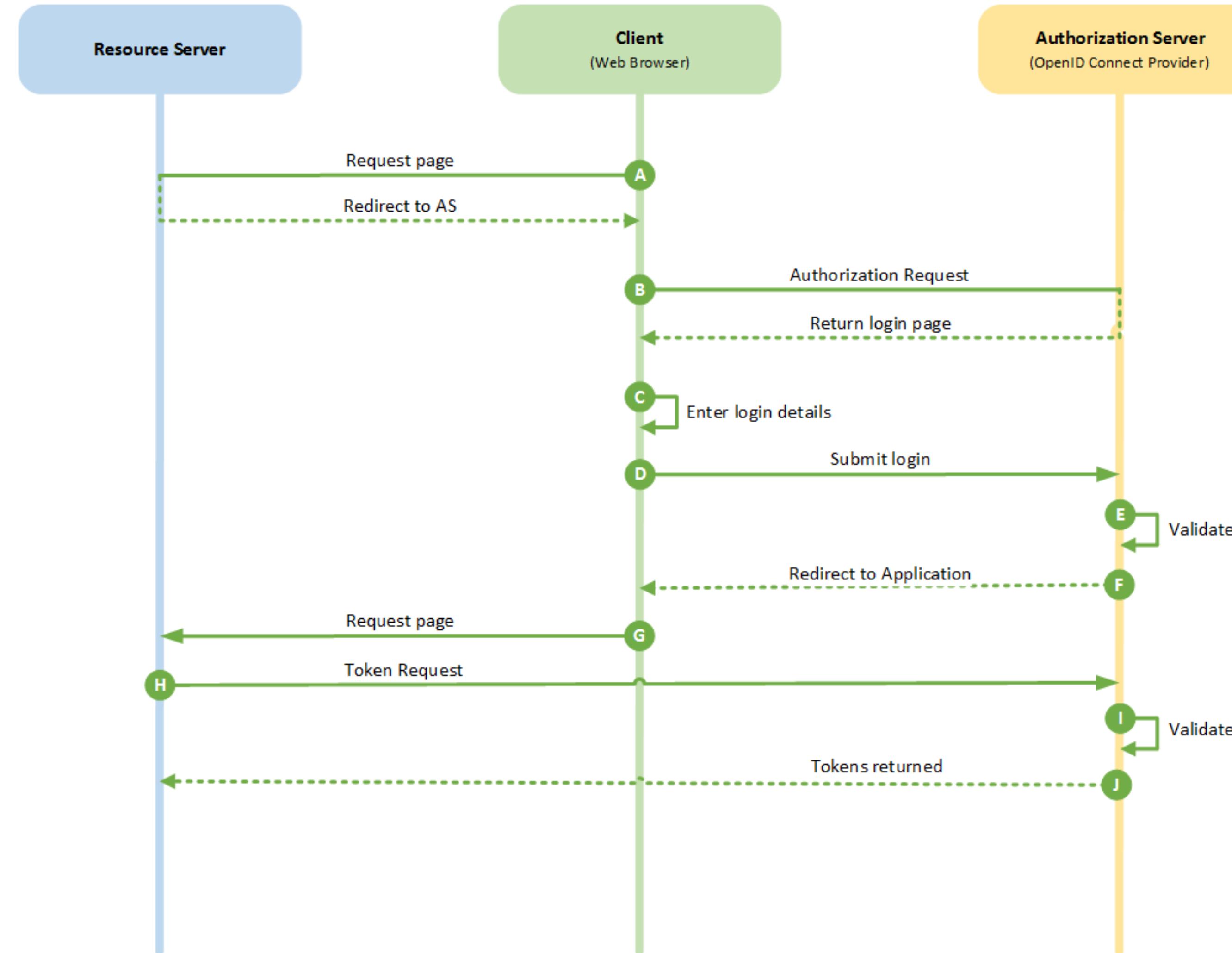
# Data flow diagrams

**Level 0 DFD**

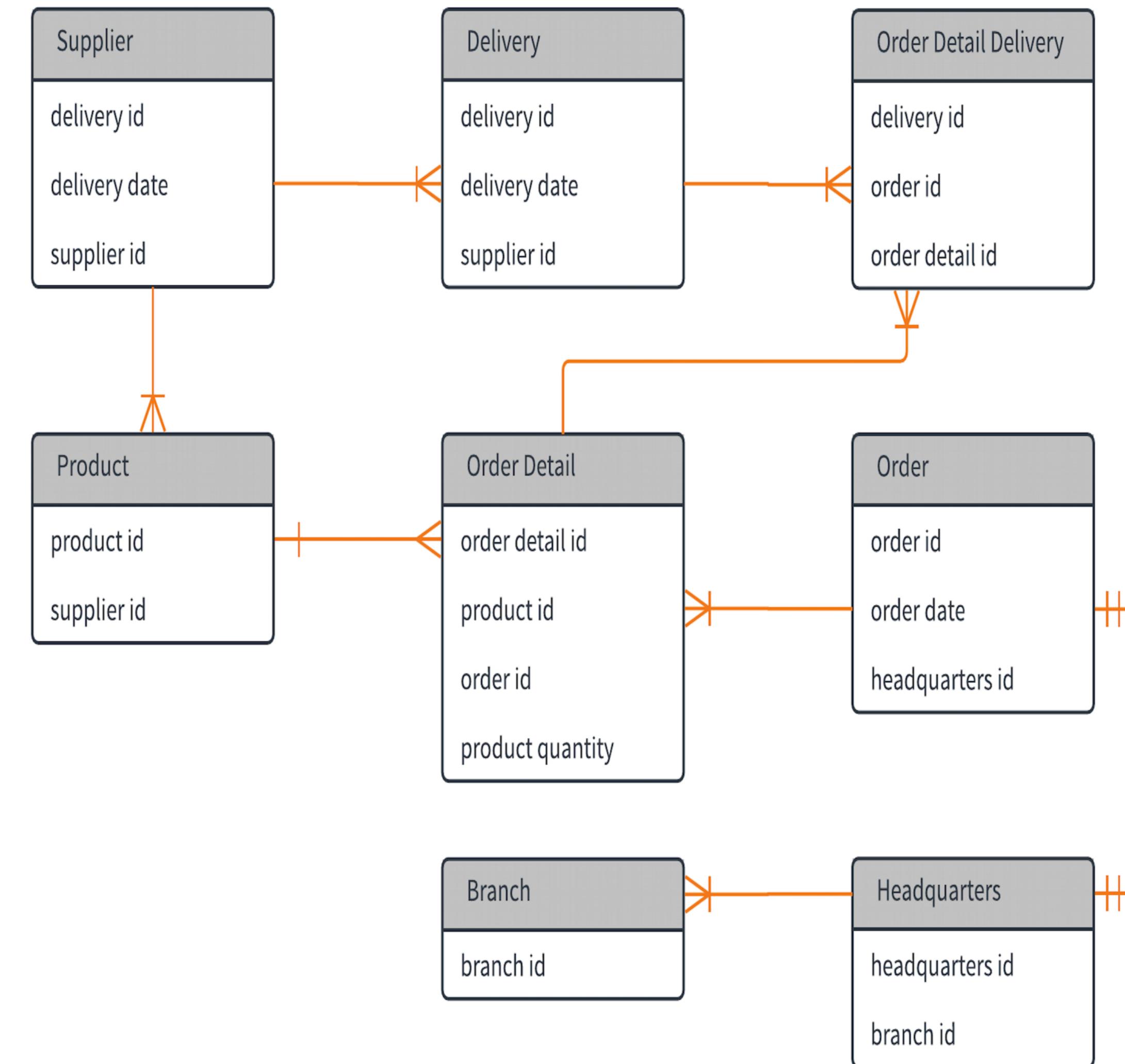


# Sequence diagram

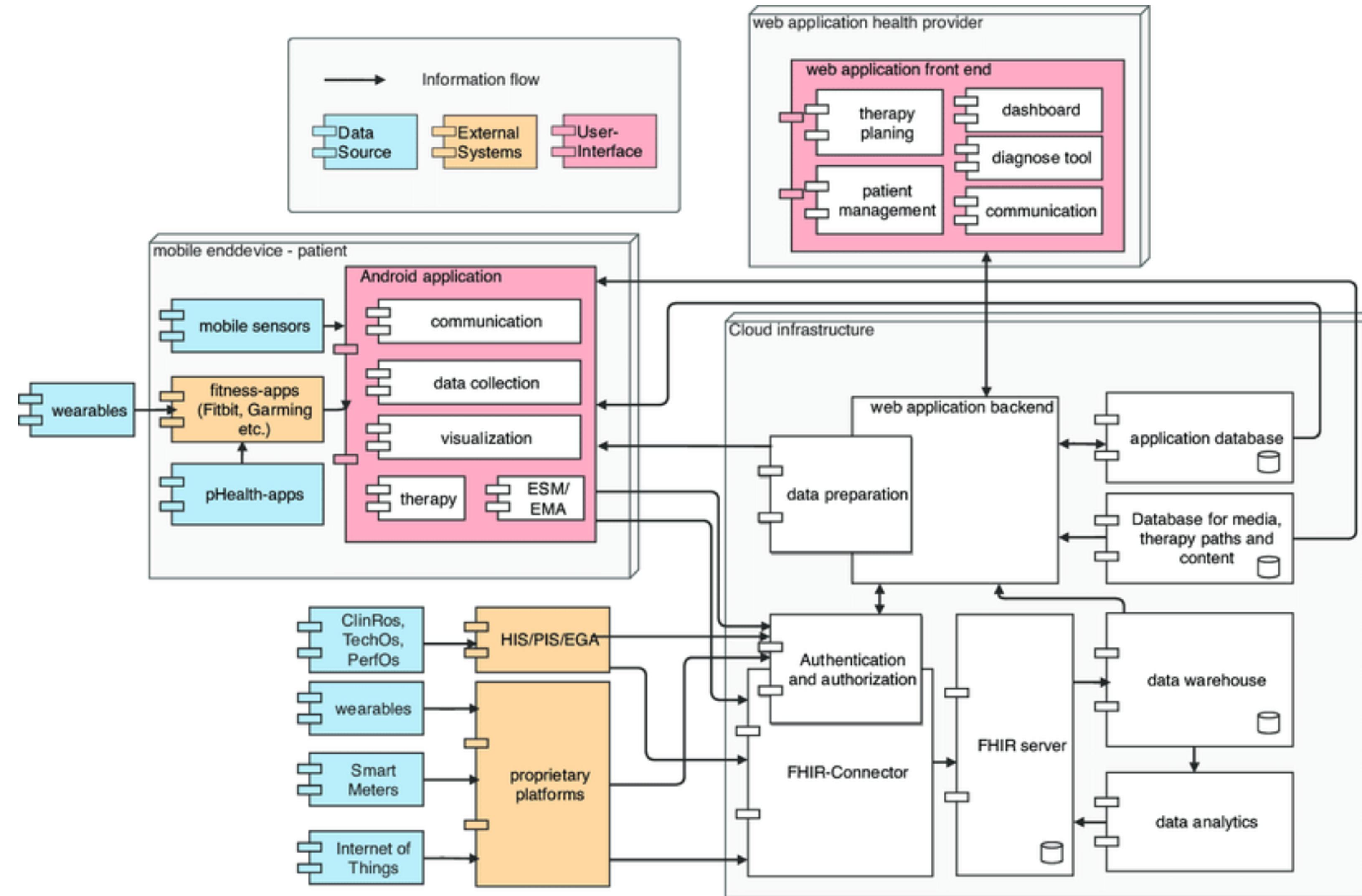
OpenID  
protocol



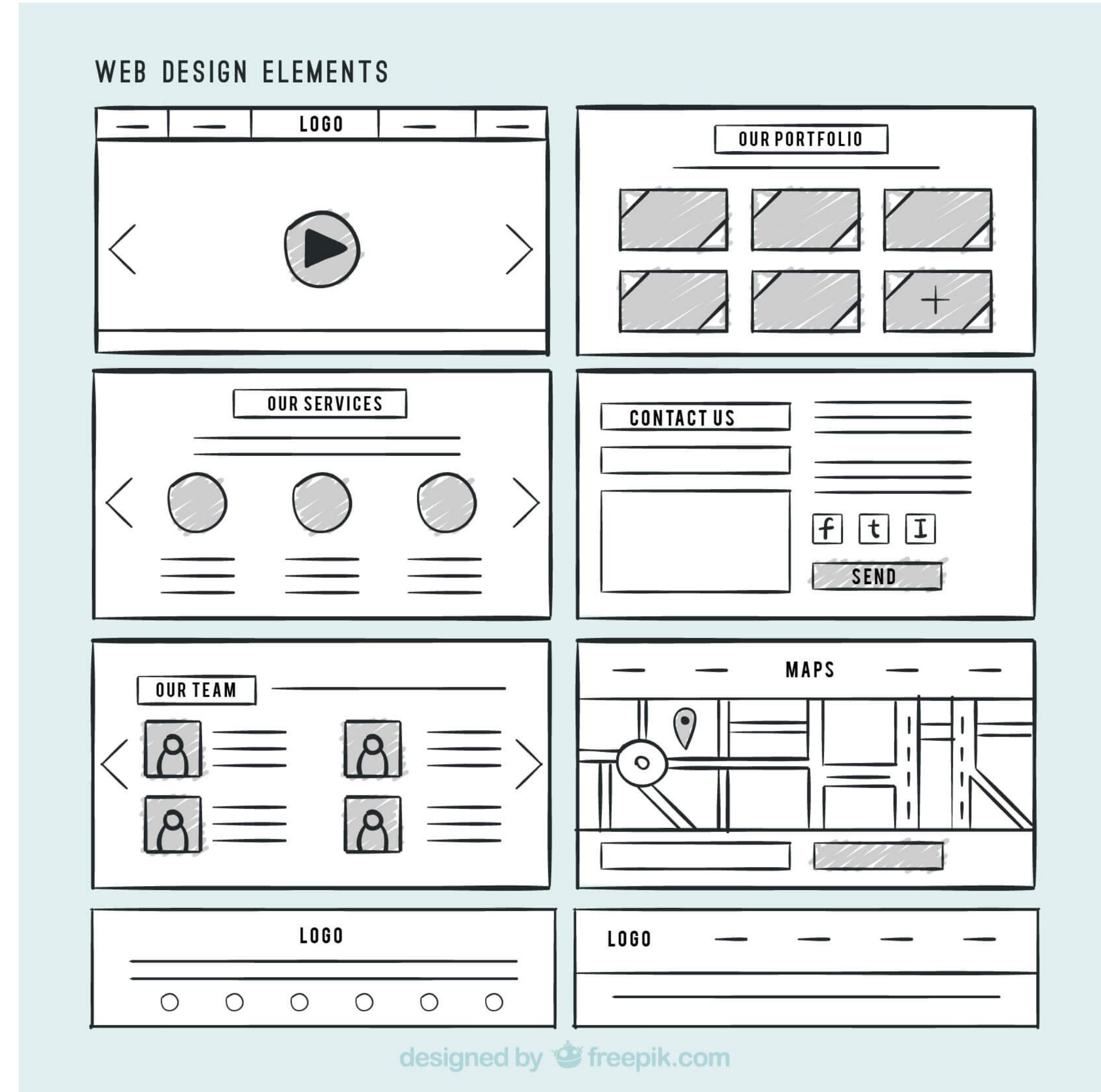
# Entity-Relationship diagram



# Component architecture



# UI mockups



# Pseudo code

---

**Algorithm 1** My algorithm

---

```
1: procedure MYPROCEDURE
2:   stringlen  $\leftarrow$  length of string
3:   i  $\leftarrow$  patlen
4:   top:
5:     if i  $>$  stringlen then return false
6:     j  $\leftarrow$  patlen
7:   loop:
8:     if string(i) = path(j) then
9:       j  $\leftarrow$  j - 1.
10:      i  $\leftarrow$  i - 1.
11:      goto loop.
12:    close;
13:    i  $\leftarrow$  i + max(delta1(string(i)), delta2(j)).
14:    goto top.
```

---

# Course modus

- Inverted classroom
- Some instruction - tools, artifacts of design, systems
- Building software and critical review
- Work in teams
- Communication will be via Canvas announcements
- Use the discussion board for class interactions
- Think “collaboration” instead of “competition”.