

An Interactive End-To-End Machine Learning Platform

Implementation Report

April 18, 2021

Requirements Specification	Murat Kurnaz
Design	Mustafa Enes Batur
Implementation	Ömer Erdiñç Yağmurlu
QA / Testing	Tarek Gaddour
Final	Atalay Donat

Contents

1	Tools	3
1.1	VS Code	3
1.2	PlantUML	3
1.3	MikTEX	3
1.4	npm	3
1.5	create-react-app	3
2	Challenges	4
2.1	Client	4
2.1.1	Testing Clients	4
2.1.2	Sensor Data Collection	4
2.2	Workspace Management	5
2.3	Model Management	5
2.4	Auth	5
3	Statistics	6
3.1	client	6
3.2	workspace-management	6
3.3	model-management	6
3.4	auth-management	6
3.5	Total	6
4	Changes from Design	7
4.1	workspace	7
4.2	client	8
4.3	model	9
4.4	auth	10
5	Functional Requirements Coverage	11
5.1	Mandatory Requirements	11
5.2	Optional Requirements	11
6	Planned Schedule	13
7	Actual Schedule	14
8	Unit Tests	15

1 Tools

1.1 VS Code

Code Editor

1.2 PlantUML

Charting software

1.3 MikTEX

Latex compiler and package manager

1.4 npm

node package manager

1.5 create-react-app

an intuitive command line scaffolding application easing the development of react applications

TBD, ADD MORE STUFF HERE, PYTHON ETC

2 Challenges

2.1 Client

2.1.1 Testing Clients

The front-end is written in React and is composed of presentational components (components), stateful components (containers) and hooks. In separating presentational and stateful components from one another we wanted to ease testing. Although testing presentational components took place without a problem using Jest.js snapshots, render tests and some simple consistency tests, stateful containers were harder to test, since they required extensive mocking of React's hook and lifecycle events.

2.1.2 Sensor Data Collection

Browser Inconsistencies

from <https://github.com/PSE-TECO-2020-TEAM1/client/issues/5#issuecomment-817308653>

There are two main APIs on sensor access in browsers right now, namely the Sensors API, which is incorporated into the web standard and supported by 71.03% of all users worldwide.

On the other hand there is the legacy DeviceMotionEvent API, which was an experimental technology designed before the aforementioned Sensors API was drafted. It is supported by 94.97% of users worldwide (albeit with handicaps).

In this application, we are using the new standard Sensors API, which is only supported by Chrome and Chromium based browsers like Edge, Brave etc. for now. Apple refuses to implement the new API citing privacy concerns, there is no information on why Firefox doesn't implement it. Since on Apple platforms all browsers from all vendors use the Safari WebView, this new API doesn't work at all on Apple devices.

The DeviceMotionEvent API is unfortunately not suitable for use at all. All three different major browsers (Chrome, Safari and Firefox) have a different understanding of what coordinates they return and have no documentation of which units they return the data in.

We've tried to use a polyfill with the branch sensors-polyfill, but were getting totally different results with different browsers (and with firefox totally broken results. Because

of this reason we've decided not to support Apple users at all.

Magnetometer Originally we wanted to support Magnetometer sensor too, since it was (supposedly) supported by the browsers we were targeting (on caniuse.com).

The image shows two browser compatibility tables side-by-side. The left table is titled 'Magnetometer' and the right table is titled 'Magnetometer API'. Both tables have a header row with browser names: IE, Edge, Firefox, Chrome, Safari, Opera, and Safari iOS. Below the header is a grid of colored boxes representing compatibility status and version ranges. The colors are red (no support), green (support), and grey (partial support). The tables also include filters for 'Current aligned', 'Usage relative', 'Date relative', and 'Filtered'.

IE	Edge	Firefox	Chrome	Safari	Opera	Safari iOS
	12-18		4-57		10-53	
6-10	79-89	2-86	58-66	3.1-13.1	54-72	3.2-14
11	90	87	90	14	73	14
		88-89	91-93	TP		

IE	Edge	Firefox	Chrome	Safari	Opera	Safari iOS
	12-18		4-55		10-42	
6-10	79-89	2-86	56-89	3.1-13.1	43-72	3.2-14
11	90	87	90	14	73	14
		88-89	91-93	TP		

After implementing the sensor, however, we've noticed how no matter what we've tried, we weren't getting any data, and after more investigations we've discovered that the 'Magnetometer Sensor' and the 'Magnetometer Sensor API' are two different entities separate from each other, and dropped support for magnetometers.

2.2 Workspace Management

2.3 Model Management

2.4 Auth

3 Statistics

3.1 client

Lines of code	TBD
Test coverage	TBD
Number of commits	RBD

3.2 workspace-management

3.3 model-management

3.4 auth-management

3.5 Total

4 Changes from Design

4.1 workspace

4.2 client

4.3 model

4.4 auth

5 Functional Requirements Coverage

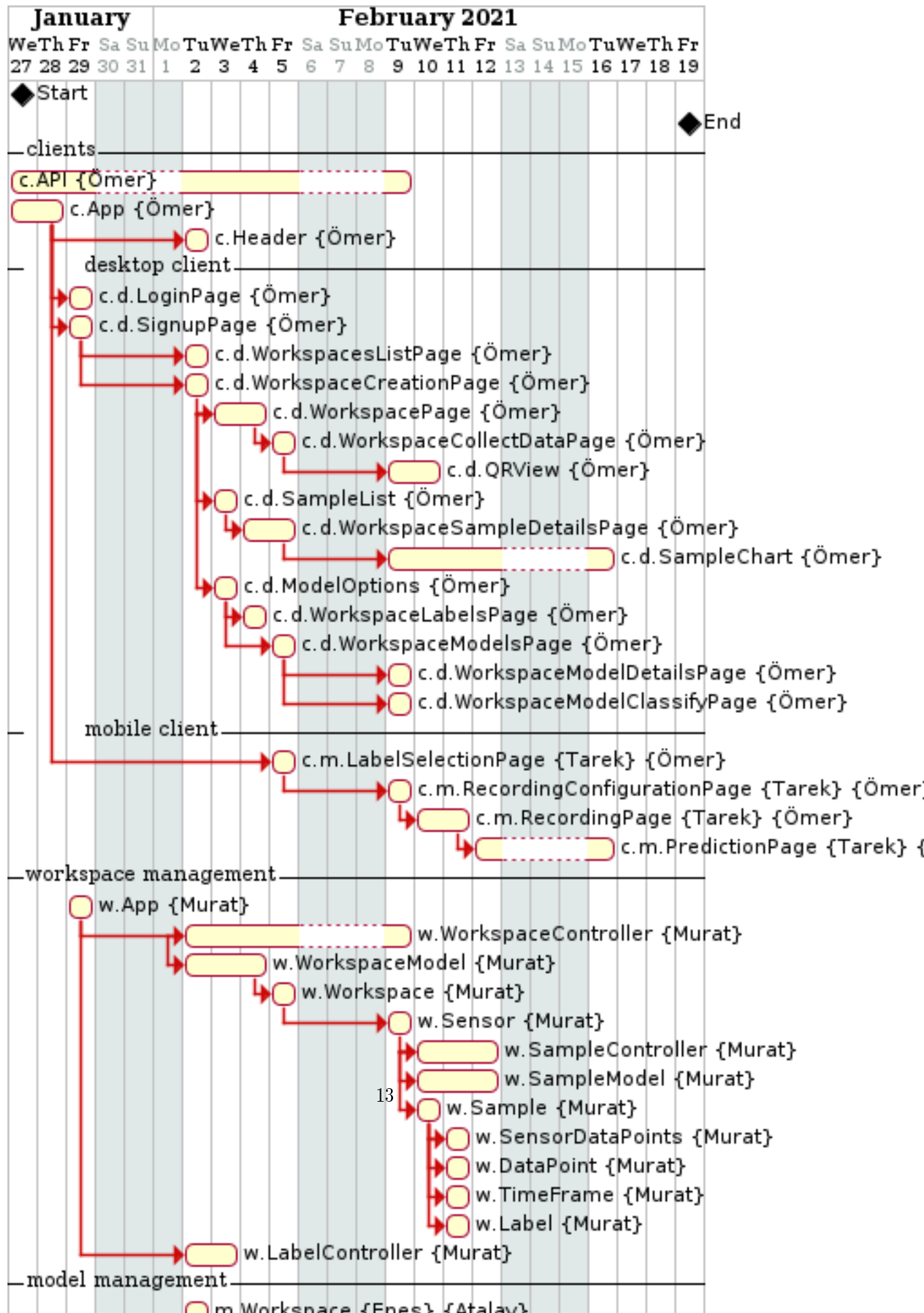
5.1 Mandatory Requirements

Number	Requirement Name	Implemented?	Notes
NUM	Here comes the name	YESNOPARTIALLY	

5.2 Optional Requirements

Number	Requirement Name	Implemented?	Notes
NUM	Here comes the name	YESNOPARTIALLY	

6 Planned Schedule



7 Actual Schedule

TBD: CAN EVERYONE UPDATE THE GANNT CHART IN THE PLAN FOLDER
WITH WHAT THEY'VE DONE

8 Unit Tests

TBD