

Kuressaare Ametikool
IT ja ettevõtluse õppesuund
Noorem tarkvaraarendaja

Keijo Raamat
BLOGI 'KÜPSETAJA TÖÖLAUD'
Lõputöö

Juhendaja:
Janek Mander

Kuressaare 2020

Sisukord

Sissejuhatus.....	3
1. Töö eesmärk ja olulisus	3
2. Tehniline kirjeldus	4
2.1. Olulisemate funktsionaalsuste detailne spetsifikatsioon	4
2.1.1. Kirjutaja sisse logimine.....	4
2.1.2. Postituse lisamine	4
2.1.3. Postituse muutmine.....	5
2.1.4. Postituse kustutamine	5
2.1.5. Postituse kuvamine	5
2.2. Koodi visuaalne kujundamine	5
2.3. Andmebaasi kirjeldus	6
2.3.1. Tabeli <i>Users</i> kirjeldus	6
2.3.2. Tabeli <i>Posts</i> kirjeldus.....	6
3. Praktilise teostuse kirjeldus	8
3.1. Töövahendid	8
3.2. Töövõtted	8
3.2.1. Arenduse voog (<i>flow</i>).....	9
3.3. Veebilehe reageerivus (<i>responsive web</i>).....	9
4. Meeskonna koosseis, ülesannete jaotuskava	11
4.1. Koosseis.....	11
4.2. Ülesannete jaotus.....	11
5. Teostamise ajakava.....	12
6. Tulemuste analüüs	13
Kasutatud allikad	14
Lisad.....	15
Kanban tahvel	15
Aja logimine	16
Visuaalne plaan	17

Sissejuhatus

Käesolev projekt ja selle dokumentatsioon on koostatud õppeasutuse Kuressaare Ametikool Noorem tarkvara-arendaja stuudiumi lõputööna. Tegu on ajaveebiga (*blog*)¹, milles käesoleva töö autor peab järke oma hobi, küpsetamise, üle. Selle ajaveebi projekteerimisel ja ehitamisel on üritatud rakendada tarkvara välearenduse² meetodeid (*agile software development*).

Kasutatud on hüperteksti markeerimiskeelt HTML'i, peamise programmeerimiskeelena PHP-d ja vähesel määral JavaScripti. Andmete hoiustamiseks on kasutatud Postgresql andmebaasi, koodi versioneerimiseks Git'i versiooni haldustarkvara. Projekti praktiline osa on majutatud Heroku keskkonda (<http://kypsetaja-toolaud.herokuapp.com/>).

1. Töö eesmärk ja olulisus

Töö esmaseks eesmärgiks on demonstreerida lugejaile stuudiumi jooksul omandatud oskusi ja teadmisi. Andmebaasi andmete lisamine, pärimine, muutmine ja kustutamine on iga veebilehe alus. Ja kui on tegemist andmetega siis on äärmiselt oluline nendega turvaliselt ringi käia.

Loodud ajaveebi saab kirjutaja lisada postitusi koos fotodega. Postituste vaatlemine on võimaldatud kõigile. Muutmiseks peab aga kasutaja olema tuvastatud.

Töö tulemusena valmiv ajaveeb on oluline seetõttu, et sellega saab kirjutaja oma hobi hõlpsamini hallata ja struktureerida.

¹ Siin ja edaspidi sulgudes inglise keelsed vastsed.

² Eestikeelne tõlge Sõnaveebist: <https://sonaveeb.ee/search/unif/dlall/dsall/agiilne/1>

2. Tehniline kirjeldus

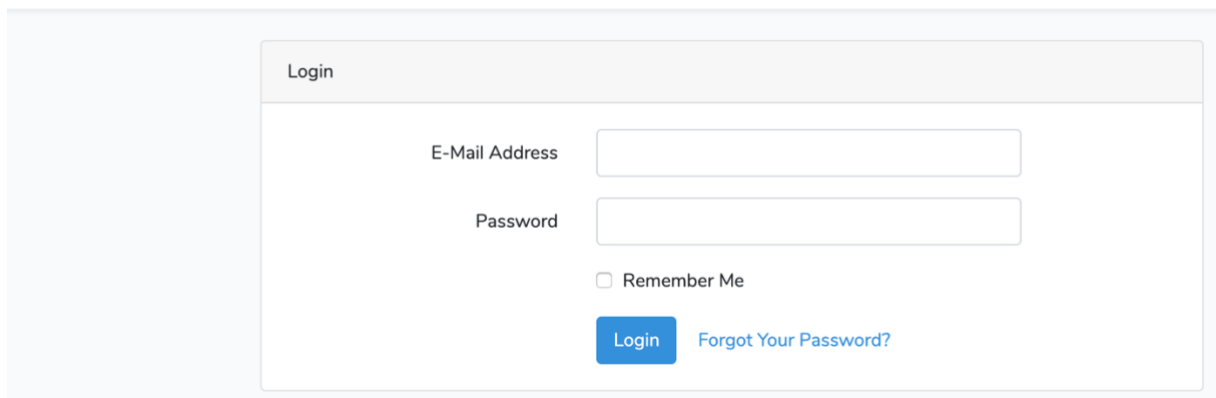
Ajaveeb on ehitatud kasutades mudel-vaade-kontroller (*model-view-controller, MVC*) arhitektuuri mustrit.

2.1. Olulisemate funktsionaalsuste detailne spetsifikatsioon

2.1.1. Kirjutaja sisse logimine

Selleks, et kindlustada ajaveebi ainult ühe, konkreetse, inimese postituste kirjutamine ja haldamine peab kirjutaja sisestama oma emaili aadressi ja parooli logimise vormi (pilt 1). Vormist saadavaid andmeid võrreldakse andmebaasis olevatega. Sobivuse korral kasutaja logitakse sisse. Andmebaasi kasutajate (*users*) tabelis hoiustatakse nime (*name*), emaili aadressi ja parooli räsi (*hash*). Reaalset parooli turvalisuse eesmärgil kuhugi ei salvesta. Logimisel vaid kontrollitakse, kas vormi sisestatud paroolis saab krüpteerides sama räsi, mis andmebaasis on.

kypsetaja_toolaud



Pilt 1. Sisse logimise vorm

2.1.2. Postituse lisamine

Postituste kirjutamine toimub loomise vaates */posts/create*. Postitusel on kohustuslikuks osaks pealkiri ja postituse sisu. Lisaks on võimalik kategoriseerida postitust, hinnata küpsetist, millest postitus räägib ja lisada foto küpsetisest. Postitus salvestatakse andmebaasi tabelisse nimega *posts*. Kui postitusele on lisatud pilt, siis see salvestatakse *uploads/images/* kausta ja andmebaasi salvestatakse viit (*link*) pildi failini. Igale postitusele genereeritakse unikaalne vöti – *id*.

2.1.3. Postituse muutmine

Postitust saab muuta muutmise vaates *posts/<postituse id>/edit*. Viide sellele vaatele kuvatakse postituse lugemise vaates ainult sisse logitud kasutajale. Kui sellele viitele peaks sattuma mitte logitud kasutaja siis kuvatakse teadet sisse logimise vajalikkusest.

Muutmise vaade on sisestamise vaate nii-öelda kloon. Erisuseks on see, et vormi sisestuse väljad on täidetud andmebaasist pärit postituse andmetega.

2.1.4. Postituse kustutamine

Postituse kustutamiseks kasutatakse viitel */posts/<postituse id>* hüperteksti edastusprotokolli meetodit DELETE. Nupp, mis sellele viitele DELETE meetodit rakendab kuvatakse ainult logitud kasutajale. Samuti on viitele lubatud erinevaid meetodeid rakendada ainult logitud kasutajal.

2.1.5. Postituse kuvamine

Ava vaatel, viide /, kuvatakse kõikide postituste lühidaid eelvaateid. Iga eelvaade koosneb postituse lisamise ajast, pealkirjast, esimesest 250 tähemärgist ja fotost.

2.2. Koodi visuaalne kujundamine

Vabalt kujundatavad (*free-form*) programmeerimiskeeltega programmeerides on kirjutajal võimalik kasutada tühikuid ja rea vahetusi meelevaldselt. PHP on saanud oma alguse C keelest ja on pärinud C-st ka vabalt kujundatavuse. (Vikipeedia vabalt kujundatavatest keeltest, 2018).

Kuna programmeerimine on lisaks arvutile instruktsioonide andmisele ka inimestele enda sammude kirjeldamine, siis on vajalik teatud kord lähtekoodi visuaalsel kujundamisel. Selleks on välja kujunenud mõned treppimise stiilid, näiteks K&R, Horstmann, GNU ja Allman. (Vikipeedia treppimise stiilidest, 2020).

Kasutan lähtekoodi koostamisel Allmani stiili. Eelistan Allmani seetõttu, et funktsioonide pealkiri on selgelt eraldatud funktsiooni sisust. Seega kogu kood tundub mulle kergemini loetav (pilt 2).

```

29     ... public function create()
30     ... {
31     ...     ... return view('posts.create');
32     ... }
33

```

Pilt 2. Treppimise näide

2.3. Andmebaasi kirjeldus

Andmebaas koosneb kahest tabelist: *Users* ja *Posts*. Kuna konkreetse ajaveebi puhul on tegu ühe kasutajaga lehega, siis ei ole seoseid nende tabelite vahel loodud. Tabelid on automaatselt genereeritud Laraveli sisse ehitatud funktsioonide poolt. Viimane tingib seda, et kõik tulbad tabelites ei pruugi alati kasutust leida.

2.3.1. Tabeli *Users* kirjeldus

Tabel *Users* koosneb kaheksast tulbast (*column*) (pilt 3). Kasutaja lisamisel täidetakse tulbad *name*, *email*, *password* kasutaja antud andmetega. Tulpade *created_at* ja *updated_at* jaoks genereerib Laravel ajatempli (*time stamp*) vastavalt loomise ja uuendamise ajale. Tulpa *email_verified_at* lisatakse ajatempel, kui kasutaja on kinnitanud oma emaili aadressi. Tulpa *id* tulevad andmed andmebaasi sisemisest funktsioonist, mis nummerdab kõik read unikaalsete numbrita. (Laraveli dokumentatsioon, 2020)

Kuna tegu on ühe kirjutaja ajaveebiga, siis on registreerimine ja emaili verifitseerimine eemaldatud. Tabelis olevad kasutust mitte leidvad tulbad on jäetud sisse selleks, et projekti edasisel arendamisel ei peaks suuri muudatusi sisse tooma.

id	name	email	email_verified_at	password	remember_token	created_at	updated_at
1	Postitaja	1234@12		\$2y\$10\$.WPr		07:40 24.05.20	07:40 24.05.20
<Auto							

Pilt 3. Tabel *Users*

2.3.2. Tabeli *Posts* kirjeldus

Tabel *Posts* koosneb kaheksast tulbast (pilt 4). Postituse lisamisel täidetakse tulbad *title*, *body*, *category*, *image_url* ja *rating* kasutaja antud andmetega. Tulpade *created_at* ja *updated_at*

jaoks genereerib Laravel ajatempli (*time stamp*) vastavalt loomise ja uuendamise ajale. Tulpa *id* tulevad andmed andmebaasi sisemisest funktsioonist, mis nummerdab kõik read unikaalsete numbritega. (Laraveli dokumentatsioon, 2020)

	id	title	body	category	image_url	rating	created_at	updated_at
1	1	Nisuleib	Saksa	leib-sai	uploads/	5	07:40 24.05.20	07:40 24.05.20
2	2	Rulli ots	Pea	nipp	uploads/	4	10:58 24.05.20	10:58 24.05.20

Pilt 4. Tabel *Posts*

3. Praktilise teostuse kirjeldus

3.1. Töövahendid

Töövahendite hulk (*stack*) kogunes küllatki pikaks:

- Koodi kirjutamiseks Microsofti Visual Studio Code.
- Töö planeerimiseks Miro board
- Lehe kuju visualiseerimiseks InVision Studio
- Koodi versioneerimiseks Git
- Koodi hoidlaks GitHub
- Kohaliku andmebaasi jooksumiseks Docker
- Tugiraamistik koodi kirjutamiseks Laravel
- Visuaalsed komponendid Bootstrap

3.2. Töövõtted

Koodi kirjutamisel proovin valida muutujate ja funktsioonide nimetused selliselt, et nad kirjeldaksid kasutatavat sisu niipalju kui võimalik. Lähtun ühe tarkvara välearenduse manifesti looja, Robert C. Martini (Tarkvara välearenduse manifest, 2001) ideedest:

- Kood peab olema loetav nagu proosa.
- Pikkade kommentaaride asemel muuda kood loetavamaks.

(Martin, Robert C., 2008)

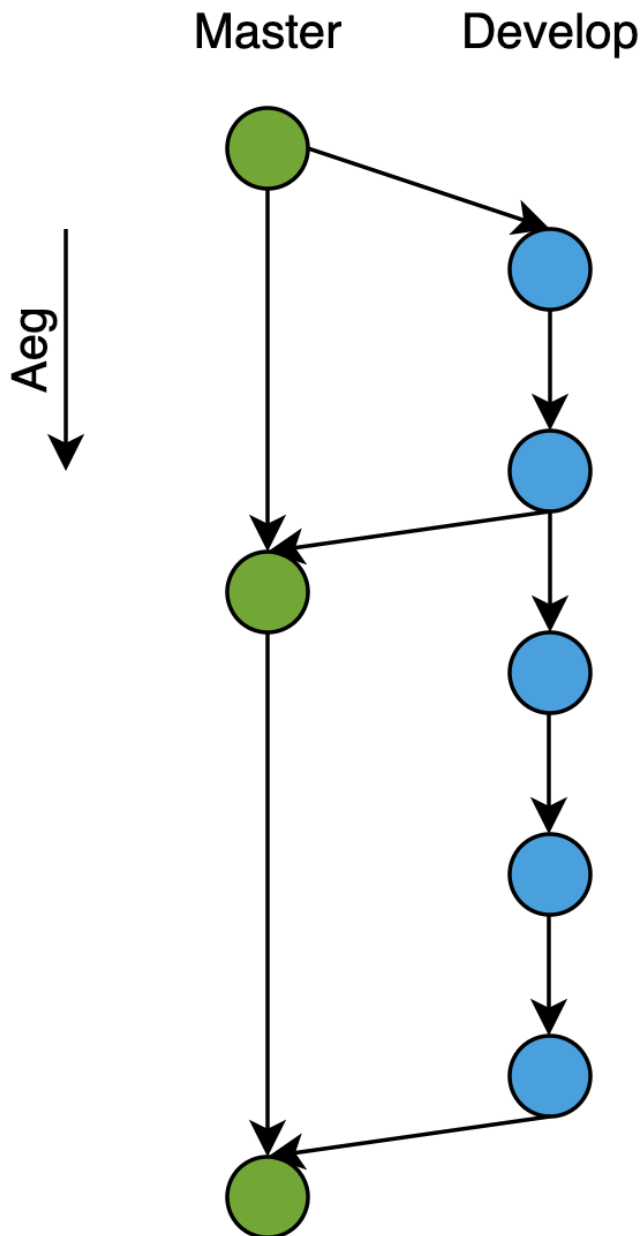
Lisaks võimalikult sisukatele nimedele koodis kirjutatan tihti ja kirjeldavaid git pühendusi (*git commit*, *commit*). Sisukad ja iga väiksemagi funktsionaalsuse lisandumise korral tehtud *commit* on justkui dünaamiline kommentaar koodile. Kasutan töös VS Code'i koos laiendusega GitLens. GitLens kuva iga koodi rea juurde viimatisel *commiti* sisu (pilt 5). Kui koodi kirjutatud kommentaarid muutuvad väga kergelt koodiga mitte kokku kuuluvaks (näiteks unustatakse koodi muutes kommentaari muuta), siis giti repositooriumis (*repository*) säilib kogu ajalugu.

```
36 <div class="p-1 form-group">
37   <div>
38     <label for="image">Foto</label>
39     <input class="btn btn-light" type="file" id="image" name="image" >
40   </div>      Keijo Raamat, 2 days ago • added image uploading to post creating view
41 </div>
42
```

Pilt 5. Git *commit* kui kommentaar

3.2.1. Arenduse voog (*flow*)

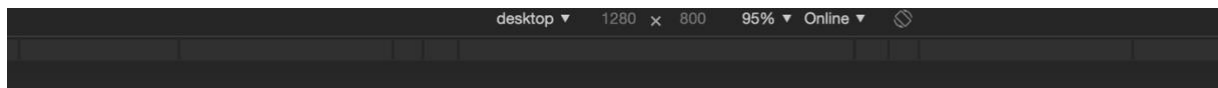
Tulenevalt meeskonna väiksusest ja projekti lihtsusest kasutan *commitimisel* elemente *GitHub flow* ja tüvel rajaneval (*trunk based*) arendusvoogu. *Master* harus hoian alati töötavat, väljastatavat (*deployable*), koodi. *Develop* harusse teen jooksvad *commitid*. Kui näen, et olen arendusega jõudnud nii kaugele, et üks ajaveebi omadus on arendatud välja ühildan (*merge*) *develop* ja *master* haru omavahel (joonis 1).



Joonis 1. Arenduse voog

3.3. Veebilehe reageerivus (*responsive web*)

Veebileidese ehitamisel kasutan Bootstrap'i teeki. Bootstrap muudab reageeriva lehe loomise küllaltki lihtsaks ja mugavaks (pildid 6 ja 7).



[Logi sisse](#)

KÜPSETAJA LOGIRAAMAT

Küpsetajast Lingid

24 05 2020 **Nisuleib juuretisega**

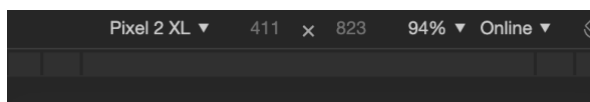
Saksa keeles

[Loe lisa](#)

Weizensauerteigbrot on eesti keeles lihtsalt sai. Kui eestikeelne nimetus sai viitab tavaliselt pehme kollaka koorikuga plönnile, mida äärmiselt tervise vaenulikuks peetakse, siis nüüd sai küpsetatud krõbeda kooriku ja pehme, arom...



Pilt 6. Töölaua vaade 1280X800 suurusel ekraanil.



[Logi sisse](#)

KÜPSETAJA LOGIRAAMAT

Küpsetajast Lingid

24 05 2020 **Nisuleib juuretisega**

Saksa keeles

[Loe lisa](#)

Weizensauerteigbrot on eesti keeles lihtsalt sai. Kui eestikeelne nimetus sai viitab tavaliselt pehme kollaka koorikuga plönnile, mida äärmiselt tervise vaenulikuks peetakse, siis nüüd



Pilt 7. Mobiili vaade 411X823 suurusel ekraanil

4. Meeskonna koosseis, ülesannete jaotuskava

4.1. Koosseis

Meeskond koosnes ühest isikust. Seetõttu võtsin ainuisikuliselt enda kanda kõik vajalikud rollid tarkvara arendamiseks.

4.2. Ülesannete jaotus

Ülesanded püstitasin ja kategoriseerisin lähtudes agiilsetest tarkvaraarenduse metoodikatest. Esmalt sõnastasin endale, milliseid omadusi soovin ajaveebis näha. Seejärel otsustasin ilma milliste omaduste ei saa tööd ajaveebiks nimetada. Nii selgitasin välja vähima elujõulise toote (*minimum viable product* ehk *MVP*). Seadsin MVP täitmiseks vajalikud omadused prioriteetsuse listis esimeste hulka.

Kuna kogu arendus toimub väga lühikeses ajaraamis. Kogu projekti pikkuseks on nõutud minimaalselt 40 tundi. Siis otsustasin viia läbi arenduse Kanbani meetodi võtteid jälgides. Kanbani korral võetakse töösse nimekirjast (*product backlog*) järgmine ülesanne kohe kui tekib vabu ressursse töö teostamiseks (lisa Kanban tahvel).

5. Teostamise ajakava

Töö teostamiseks jagasin etteantud aja ligikaudu kolmeks: 1/3 planeerimiseks, 1/3 programmeerimiseks ja 1/3 dokumenteerimiseks. Kuna tegu oli õppetöö protsessiga otsustasin jaotada tööaja pikema perioodi peale. Karme ajalisi reegleid endale ei seadnud. Eesmärk oli kasutada olemasolevat aega võimalikult rentaablilt ja õppida võimalikult palju. Kronoloogiline töötegemise graafik on ära toodud lisas Aja logimine.

6. Tulemuste analüüs

Tulenevalt sellest, et jaotasin töötegemise pikema aja peale laiali, suutsin õppides projekteerida ja arendada. Igapäevased ja järjepidevad nn. tööampsud võimaldasid süveneda erinevatesse aspektidesse ning leida iseseisvalt lahendusi ettetulnud probleemide.

Programmeerides on aga probleemid igapäevased nähtused ning oskus lahendusi leida kriitilise tähtsusega.

Kõiki algselt planeeritud ülesandeid ei jõudnud küll etteantud aja jooksul ära teha, kuid töö käigus omandatud oskused võimaldavad tegemata jäänud ülesandeid vägagi kiiresti lahendada. Näiteks lisada kategooriad eraldi tabelisse ja neid eraldi kuvada oleks nüüd küllaltki lihtne.

Projekti läbiviimisel kinnistasin mitmeid teoreetilisi teadmisi, mida olin varasemalt koolis õppinud, näiteks arhitektuuri mustri MVC kasutamine.

Kokkuvõtvalt võib tõdeda, et see iseseisvalt tehtud töö oli väga õpetlik, julgustav, kuid andis samas mõista, et ainult üksinda töötamise vorm ei ole mulle sobiv ja eelistan meeskonnatööd.

Korduvalt tundsin, et tarkvara arendamine oleks olnud olulisemalt kiirem kui oleks olnud võimalus kellegagi ideid nii-öelda põrgatada.

Kasutatud allikad

Tarkvara välearenduse manifest: Agile manifesto (2001). Vaadatud 26.05.2020
<https://agilemanifesto.org/iso/et/manifesto.html>

Laraveli dokumentatsioon: Laravel docs (2020). Vaadatud 24. 05 2020
<https://laravel.com/docs/7.x>

Martin, Robert C. (2008). *Clean Code: A Handbook of Agile Software Craftsmanship*. New Jersey: Prentice Hall

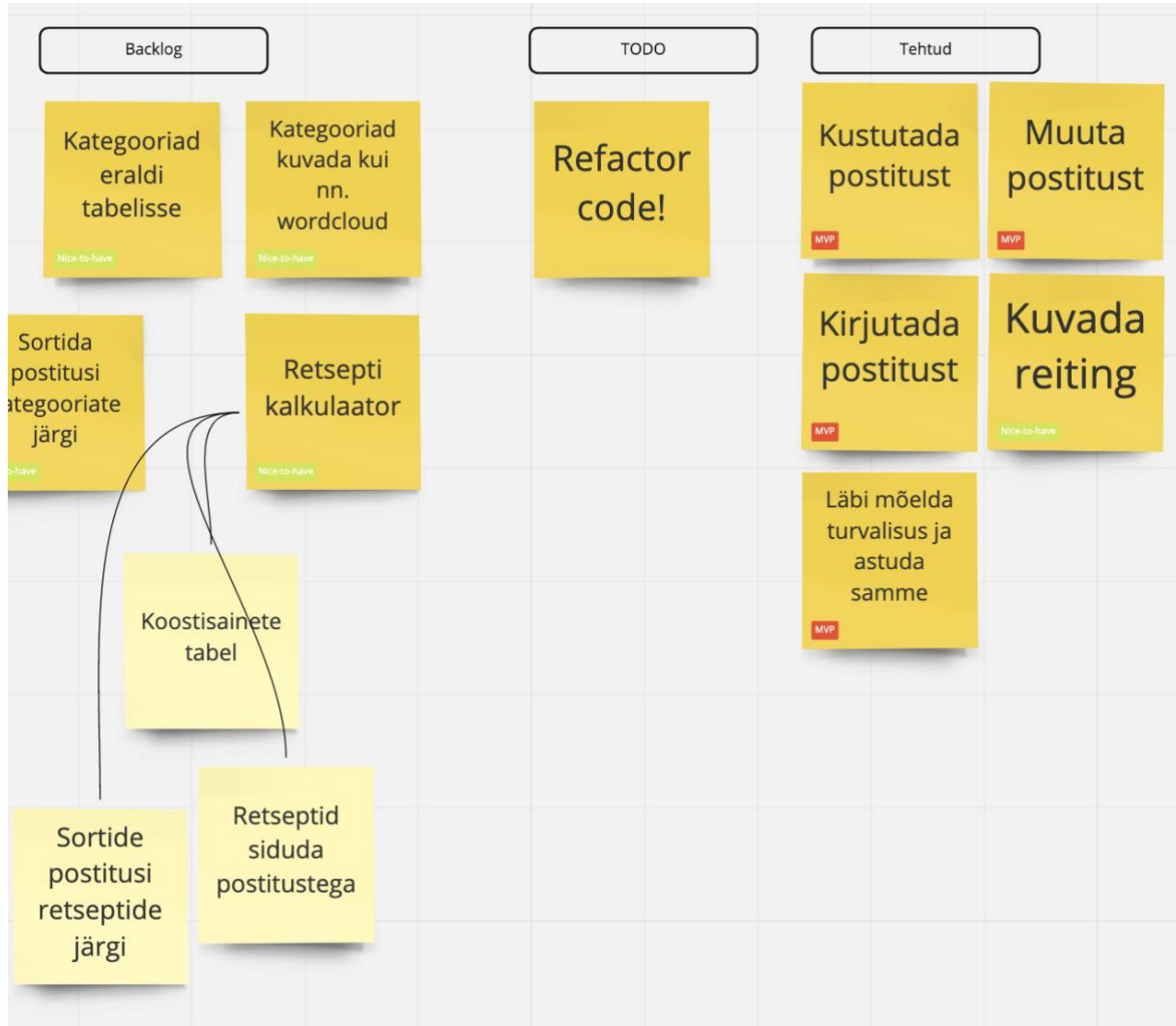
Vikipeedia treppimise stiilidest: Indentation style (2020). Vaadatud 25.05.2020
https://en.wikipedia.org/wiki/Indentation_style

Vikipeedia vabalt kujundatavatest keeltest: Free-form language (2018). Vaadatud 25.05.2020 https://en.wikipedia.org/wiki/Free-form_language

Lisad

Kanban tahvel

https://miro.com/app/board/o9J_krgQ_BE=



Aja logimine

Aja logimiseks kasutasin lihtsat tabelit. Tabelisse kirjutasin kuupäeva millal, probleemi, mille kallal ja tunnid kui kaua töötasin (tabel 1).

Kuupäev	Probleem	tunnid
02.05.2020	Idee jaotamine detailsemateks tükideks	2
	Visuaalne eskiis ja fontide valik	4
03.05.2020	Backlog koostamine	1
	MVP selgitamine	1
04.05.2020	Featuuride jagamine 'nice-to-have' ja MVP vahel	1
05.05.2020	Arenduskeskonna ülesseadmine (lokaalsed tööriistad ja deploy flow)	2
	Migrations	1
06.05.2020	Programmeerimine	1
07.05.2020	Programmeerimine	2
08.05.2020	Programmeerimine	1
11.05.2020	Ülesannete ülevaatamine	1
12.05.2020	Programmeerimine	2
13.05.2020	Programmeerimine	2
15.05.2020	Programmeerimine	2
16.05.2020	Programmeerimine	6
17.05.2020	Programmeerimine	5
18.05.2020	Programmeerimine	1
19.05.2020	Programmeerimine	1
20.05.2020	Programmeerimine	2
22.05.2020	Kirjaliku osa kirjutamine	6
23.05.2020	Programmeerimine	3
24.05.2020	Kirjaliku osa kirjutamine	7
25.05.2020	Kirjaliku osa kirjutamine	8
	Kokku	62

Tabel 1

Visuaalne plaan

<https://keijoraamat896885.invisionapp.com/prototype/ckanzp15h00d3ga0163iiiwq7/play>

