



Mooduli nimetus: Tarkvaraarendus AI-ajastul

Mooduli lühikirjeldus:

Tarkvaraarendus on viimase kolme aastaga põhjalikult muutunud: AI-tööriistad kirjutavad koodi kiiremini kui inimene ning arendaja roll liigub üha enam kavandamise, AI juhendamise ja tulemuse kontrollimise suunas. Tarkvarafirmades on tekkinud uus töövorm - agentne tarkvaraarendus (agentic engineering) -, kus arendaja juhib AI-agente, kes teevad töö tema plaani järgi ära. Seda oskust valdavad alles vähesed ja just seda me selles moodulis õpime.

Kuna arendaja töö on nüüd rohkem kavandamine, selge eneseväljendus ja koostöö, on selles ruumi väga erinevate tugevustega õpilastele - ka neile, kes ei ole end seni "arvutiinimeseks" pidanud.

Aluspõhimõtted siiski ei kao: et hinnata, kas AI loodud kood on õige ja turvaline, tuleb programmeerimist tunda sügavuti. Seepärast annab moodul mõlemad pooled - kindla programmeerimisoskuse põhja ja praktilise kogemuse AI-agentidega tarkvara ehitamisel. Kõrvale põimime kergemas vormis ka arvutiteaduse aluseid (kuidas arvuti andmeid esitab, kuidas kood käivitub, kuidas keelemudel töötab).

Mooduli tööpõhimõtted:

- "AI kirjutab, sina vastutad" - õpilane peab oskama iga koodirea selgitada ja põhjendada;
- kõigepealt käsitsi - põhimõisted (muutujad, tingimuslaused, tsüklid, funktsioonid) kirjutab õpilane ise, enne kui AI neid tema eest kirjutab; käsitsi kirjutamine ei kao ka hiljem, vaid jääb võrdluspartneriks;
- AI-luba on künnis, mitte kuupäev - AI hakkab koodi kirjutama 2. kursuse 5. tunnist, pärast seda kui õpilane oskab AI-ta ennustada lihtsa programmi väljundit;
- testid varakult - esimene test kirjutatakse koos esimese funktsiooniga (2. kursuse 3. tund) ja testid on sealt edasi igas tunnis;
- koodi ülevaatus (code review) varakult - alates 2. kursuse 6. tunnist vaatab õpilane kaaslase tööd GitHubis pull request'i kaudu üle ja kommenteerib seda;
- AI-ta lugemine igal tunnil - 10 minutit "ennusta, mida see kood teeb" enne, kui ühtegi tööriista avatakse.

Peamise programmeerimiskeelena kasutame TypeScripti, mis katab nii veebilehtede kasutajaliidese



	kui ka serveripoolse arenduse. Kõrvalpilgu heidame ka teistele keeltele (Python, Golang). AI-tööriistadest kasutame vestlusmudelit (nt ChatGPT) ja kodeerimisagenti (nt Codex); konkreetne toode valitakse igal õppeaastal.
Mooduli tüüp (Kohustuslik/Valik)	Valikmoodul
Sihtrühm	G1-G2. Soovitatav neile, keda huvitab tarkvaraarendus ja IT-valdkonnas edasiõppimine. Eelnev programmeerimiskogemus ei ole vajalik - moodul sobib ka alustajale.
Osalejate maksimaalne arv	20



Eeltingimused moodulis osalemiseks	Eelteadmisi ei eeldata. Vajalik on soov programmeerimist algusest peale selgeks õppida ja valmisolek teha praktilist tööd ning lõpuks projekt. Kasuks tuleb oma sülearvuti (koolis on olemas arvutiklass).
Mooduli kursused:	1. Sissejuhatus IT-maailma 2. Programmeerimise alused AI-ajastul 3. Tarkvaraarendus AI-ga: ideest töötava rakenduseni

1. Kursuse nimi: Sissejuhatus IT-maailma

Ainekava koostanud õpetaja(d)	Geithy Plakk, Madis Apsalon
Maht ja korraldus	20 x 70min (10 nädalat)
Kursuse toimumise aeg ja koht	Projektitööklass
Kursuse lühikirjeldus ja eesmärgid	Kursuse eesmärk on kujundada õppijas arusaam sellest, et infotehnoloogia ei tähenda ainult programmeerimist, vaid hõlmab erinevaid tehnilisi, loovaid, korralduslikke ja suhtlemisega seotud valdkondi. Õppija omandab esmased teadmised IT toimimisest, õpib lahendama lihtsamaid probleeme ning kasutama tehnoloogiat teadlikult ja vastutustundlikult. Baasteadmised infotehnoloogia peamistest teemadest.
Kursuse tulemuse kujunemise viis	Kursuse tulemus kujuneb õppeprotsessi jooksul tehtud praktiliste ülesannete, aruteludes ja rühmatöodes osalemise, iseseisvate tööde ning kursuse lõpus valmiva õpimapi põhjal, mis sisaldab tundide refleksioone.
Õpitulemused	• selgitab, mida tähendab infotehnoloogia ning millistest valdkondadest IT-maailm koosneb; • tunneb infotehnoloogiaga seotud peamisi mõisteid;



	• eristab riistvara ja tarkvara ning kirjeldab nende koostööd; • rakendab küberturvalisuse ja digihügieeni põhimõtteid; • tunneb ära levinumad ohud, näiteks õngitsuskirjad, nõrgad paroolid ja liigse info jagamise; • kasutab probleemide lahendamisel algoritmilist mõtlemist ja oskab kirjeldada tegevusi sammudena; • oskab arutleda tehnoloogia mõju, võimaluste ja riskide üle; • tunneb erinevaid IT-valdkonna ameteid ning mõistab pehmete oskuste tähtsust IT-töös; • osaleb koostöös, esitleb oma ideid ning analüüsib enda õppimist; • teostab infootsingut ja leiab lahendusi; • kasutab erinevaid digitehnoloogilisi lahendusi ülesannete täitmisel.
Vajalikud õppematerjalid ja -vahendid	Sülearvuti, internet https://progetiiger.ee/book

2.

Kursuse ülesehitus:

Teema	Tundide arv	Märkused
Sissejuhatus kursusesse, infotehnoloogiasse Riistvara ja tarkvara	2	Kursuse korraldus, Pecha Kucha Tarkvara ja riistvara selgitus, ülesande lahendamine rühmas ja iseseisvalt
Algoritmiline mõtlemine, Python, Scratch	2	Algoritmiline mõtlemine ja programmeerimise alused



		1. osa: Mis on algoritm? Igapäevaelu juhised, robotile tee juhendamine, tegevuse kirjeldamine sammudena. Plokkskeem ja pseudokood. Muutujad, tingimuslaused, kasutaja sisend, punktisüsteem ja mänguloogika. 2. osa: Esmane tutvus Scratchi või Code.org-iga. Loomes esimesed mängud (3tk) Pythoni sissejuhatus, esimese programmi loomine
Failid ja faililaiendid	2	Py kordamine Faililaiendid ja failidega töötamine
HTML/CSS Disainer - külalisesineja	2	Sissejuhatus HTML/CSS-i. Külalisesineja – UI/UX disainer
Küberturvalisus ja kvaliteet	2	Digiteenuste loomise ja haldamise turvalisuse valikud. Levinumad turvariskid digiteenuste kontekstis. Infoturbe põhimõisted. Õngitsuskirjade test Ametiprofiili tutvustus külaline–rahapesujuhtumi tõkestamine tehnoloogiaettevõttes
Andmed, Analüütik - külalisesineja	2	Kuidas jõutakse andmetest põhjendatud otsuseni ja millist rolli täidab selles AI?
Testija - külalisesineja	2	Õpimapi täiendamine Enda sobivuse testimine Kursuse sissejuhatuse refleksioon IT-maailma visualiseerimine
Tehnoloogiaettevõtte külastus	2	Wise.



Sissejuhatus moodulisse	2	
Sissejuhatus moodulisse	2	
Kokku	20	

2. Kursuse nimi: Programmeerimise alused AI-ajastul

Ainekava koostanud õpetaja(d)	Madis Apsalon
Maht ja korraldus	20 x 70 min - 10 kohtumist 2 x 70 min, lisaks iseseisev töö Iga kohtumise rütm: 10 min AI-ta lugemisharjutus ("ennusta, mida see kood teeb") - 15 min uus mõiste - 40 min ehitamine - 10 min arenduspäevik - 5 min minikaitsmine (2 õpilast selgitavad ühe rea või otsuse).
Kursuse toimumise aeg ja koht	II trimester, arvutiklass (nädalapäev ja kellaaeg täpsustamisel)
Kursuse lühikirjeldus ja eesmärgid	Kursuse eesmärk on anda õpilasele programmeerimise põhioskused: koodi lugemine, kirjutamine ja hindamine. Kursus on valdavalt praktiline - igal kohtumisel kirjutame koodi. Tunnid 1-4 kirjutab õpilane koodi ise (muutujad, andmetüübid, tingimuslaused, tsüklid) ja AI on ainult selgitaja rollis. Alates 5. tunnist käib iga ülesanne mustri käsitsi -> AI-ga -> võrdle -> testi. Käsitsi kirjutamine ei kao kursuse lõpuni, vaid jääb võrdluspartneriks. Testid tulevad sisse juba 3. tunnis koos esimese funktsiooniga ja on sealt edasi igas tunnis. 6. tunnist lisandub koodi ülevaatus: õpilased vaatavad üksteise tööd GitHubis pull request'i kaudu üle ja kommenteerivad seda - sama moodi nagu tarkvarafirmades. Kursuse lõpuks valmib igal õpilasel individuaalne mini-projekt, mille ta esitleb ja mille koodi oskab



	rida-realt selgitada. Just siin pannakse alus, millele 3. kursuse agentne arendus toetub.
--	---





Kursuse tulemuse kujunemise viis

AI-ta lugemiskontrollid igal tunnil (väljundi ennustamine, vea leidmine) - kiired ja jooksvalt hinnatavad.

Künnis 4. tunni lõpus: 10-realise programmi väljundi ennustamine paberil. Läbimine on AI-loa tingimus, teine katse 5. tunnis.

Arenduspäevik: iga tunni lõpus jagab õpilane oma AI-vestlused ja kirjutab 3 lauset - mida küsisin, mis läks valesti, mida teen järgmisel korral teisiti.

Õpetaja loeb igal nädalal 4-5 juhuslikult valitud õpilase päevikut.

Koodi ülevaatus pane alates 6. tunnist: kaaslase pull request'i kommentaarid GitHubis (kas märkas sisulisi asju, kas tagasiside on konkreetne).

Minikaitsmised: igal tunnil selgitavad 2 õpilast ühe rea või otsuse; 10 tunniga jõuab iga õpilane korra. Praktilised tööd igal kohtumisel, mis tuleb lõpetada tunnis või väljaspool seda; testid kuuluvad töö juurde alates 3. tunnist.

Kursuse lõputöö: individuaalne mini-projekt ja 3-minutiline kaitsmine, kus õpetaja valib juhusliku rea, mida õpilane selgitab.

Kursus on edukalt sooritatud, kui künnis on läbitud, praktilised tööd esitatud (vähemalt 51%), lugemiskontrollid sooritatud (vähemalt 51%) ning mini-projekt esitatud ja kaitstud.

Kuna AI suudab koodi kirjutada, ei ole esitatud koodi hulk ega ilu iseenesest hinde alus. Hinnatakse seda, mida AI õpilase eest teha ei saa: koodi mõistmist, vigade leidmist, kaaslase koodi ülevaatus ja oma lahenduse suulist põhjendamist.



Õpitulemused	Kursuse läbinud õpilane: - ennustab lihtsa programmi väljundit ilma AI-ta ja ilma koodi käivitamata; - kasutab arendaja põhitööriistu: VS Code, Git ja GitHub (repositoorium, commit, pull request); - koostab lihtsa veebilehe HTML-i ja CSS-i abil ning avaldab selle GitHub Pages'is; - kirjutab käsitsi TypeScripti põhikonstruktsioone: muutujad, andmetüübid, tehted, tingimuslaused, tsüklid ja funktsioonid; - kasutab massiive ja objekte andmete hoidmiseks ja töötlemiseks; - kirjutab oma koodile ühikteste ja kasutab silumist, et tõestada, et kood töötab; - vaatab kaaslase koodi GitHubis üle ja annab konkreetset, põhjendatud tagasisidet; - laseb AI-l sama ülesande lahendada, võrdleb seda oma lahendusega ja põhjendab, kumb on parem; - loeb AI loodud koodi, leiab selles vigu ja kirjutab testi, mis vea kinni püüab; - sõnastab ülesande AI-le nii täpselt, et tulemus on kasutatav, ja mõistab, et täpne kirjeldus on tööriist; - selgitab oma sõnadega, kuidas keelemudel töötab ja miks ta eksib, ning mille poolest erineb AI-agent vestlusrobotist; - valmistab ja esitleb iseseisvalt väikese interaktiivse rakenduse ning selgitab suvalist rida oma koodist.
Vajalikud õppematerjalid ja -vahendid	Visual Studio Code, Node.js, TypeScript Git, GitHub (repositooriumid, pull request ja koodi kommenteerimine), GitHub Pages Vestlusmudel (nt ChatGPT) - tundides 1-4 selgitaja rollis, alates 5. tunnist ka koodikirjutajana Arenduspäevik (jagatud AI-vestlused + refleksioon) Google Classroomis Google Classroom, õpetaja materjalid ja esitlused https://www.typescriptlang.org/docs/ https://developer.mozilla.org/en-US/docs/Learn Arvutiklass või oma sülearvuti



Kursuse ülesehitus:

Teema	Tundide arv	Märkused
1. Tööriistad ja esimene repositoorium Miks tarkvaraarendus on muutunud ja mis on agentne arendus Praktiline töö: esimene commit	2 x 70 min	Kursuse eesmärgid, õpitulemused, hindamiskriteeriumid ja õppekorraldus. VS Code, Git ja GitHub: konto, repositoorium, commit. Arenduspäevik algab: mida küsisin, mis läks valesti, mida teen teisiti.
2. HTML ja CSS kahel viisil Praktiline töö: oma leht käsitsi ja seejärel AI abiga, GitHub Pages'i üles	2 x 70 min	HTML-i baaselemendid ja CSS-i alused: värvid, fondid, paigutus. Võrdlus: mida AI tegi teisiti, paremini, halvemini? Siin on AI lubatud, sest tegemist on märgistuskeele, mitte veel programmeerimisega.
3. Muutujad, andmetüübid, tehted - KÄSITSI Esimene funktsioon ja esimene test Praktiline töö: funktsioon + test	2 x 70 min	Õpilane kirjutab koodi ise; AI on ainult selgitaja. Kuidas arvuti teksti ja arve mälus hoiab. Testid tulevad sisse kohe koos esimese funktsiooniga ja on sealt edasi igas tunnis - turvavõrk pannakse üles enne, kui kõiele minnakse.
4. Tingimuslaused ja tsüklid - KÄSITSI Algoritmiline mõtlemine Tunni lõpus: künnis	2 x 70 min	Õpilane kirjutab koodi ise; AI on ainult selgitaja. Künnis: ennusta paberil 10-realise programmi väljund, ilma AI-ta ja ilma käivitamata. Künnise läbinud saavad AI-loa alates 5. tunnist. Teine katse 5. tunnis.
5. Funktsioonid ja koodi loetavus Esimene AI kirjutatud kood Praktiline töö: käsitsi -> AI-ga -> võrdle -> testi	2 x 70 min	Kirjuta funktsioon käsitsi, lase AI-l sama teha, võrdle ja kirjuta mõlemale test. Kumb on loetavam ja miks? Siit edasi käib iga ülesanne selles mustris - käsitsi kirjutamine ei kao.



		Teine katse künnisel neile, kes seda vajavad.
6. Massiivid ja objektid Koodi ülevaatus (code review) GitHubis Praktiline töö: kaaslane pull request'i ülevaatus	2 x 70 min	Andmete modelleerimine: kirjelda andmed sõnadega, tee struktuur käsitsi, lase AI-l teha, kontrolli testidega. Pull request GitHubis: kuidas oma tööd ülevaatusse anda. Iga õpilane vaatab kaaslane koodi üle ja kommenteerib seda Gitis. Ülevaatus on sealt edasi osa igast tunnist.
7. Vea leidmine ja kuidas keelemudel töötab Praktiline töö: leia peidetud viga ja kirjuta test, mis selle kinni püüab	2 x 70 min	Õpetaja ja AI on koodi vigu peitnud: leia, paranda, tõesta testiga. Miks mudel eksib ja usutavalt valetab. Eelvaade agendist: mis vahe on vestlusrobotil ja agendil, kes töötab failides.
8. Promptimine kui spetsifitseerimine Praktiline töö: sama ülesanne kolme erineva täpsusega kirjeldusega	2 x 70 min	Mis muutub tulemuses, kui kirjeldus muutub täpsemaks? Esimene tõsine kohtumine ideega, et täpne kirjeldus on tööriist - see on 3. kursuse alus. Ristülevaatus jätkub: paarid kommenteerivad üksteise lahendusi Gitis.
9. Mini-projekt brauseris Praktiline töö: individuaalne interaktiivne rakendus	2 x 70 min	Rakendate seni õpitut: DOM, sündmused, kasutaja sisend. Projekt tehakse AI abiga, aga testidega ja iga rida päevikus selgitatud. Enne esitamist vaatab kaaslane projekti Gitis üle ja kommenteerib.
10. Kaitsmised ja kokkuvõte	2 x 70 min	3-minutilised kaitsmised: õpetaja valib juhusliku rea, õpilane selgitab (20 x 3 min mahub ühte tundi). Kursuse kokkuvõte, järelevastamise võimalus ja tagasiside.





Kokku	20 x 70 min (10 kohtumist)	
-------	----------------------------	--

3. Kursuse nimi: Tarkvaraarendus AI-ga: ideest töötava rakenduseni

Ainekava koostanud õpetaja(d)	Madis Apsalon
Maht ja korraldus	20 x 70 min - 10 kohtumist 2 x 70 min, lisaks iseseisev töö Iga kohtumise rütm: 10 min AI-ta lugemisharjutus ("ennusta, mida see kood teeb") - 15 min uus mõiste - 40 min ehitamine - 10 min arenduspäevik - 5 min minikaitsmine (2 õpilast selgitavad ühe rea või otsuse).
Kursuse toimumise aeg ja koht	III trimester, arvutiklass (nädalapäev ja kellaaeg täpsustamisel)
Kursuse lühikirjeldus ja eesmärgid	Kursuse eesmärk on läbida tarkvaraarenduse täitsükkel kahestes tiimides: nõuete kirjeldamine, planeerimine, kasutajaliidese ja serveripoolse arendus, testimine, koodi ülevaatus, avalikustamine ja tiimitöö. Kursuse keskmes on agentne tarkvaraarendus: õpilane õpib töötama nii, nagu töötavad tänapäeva eesindlikumad arendajad - kavandab lahenduse, annab AI-agendile täpse tööülesande, vaatab tulemuse kriitiliselt üle ja vastutab kvaliteedi eest. Käsitsi kirjutamine ja koodi lugemine ilma AI-ta jäävad ka siin igasse tundi. Kursus on üles ehitatud ühe projekti ümber, mida arendatakse esimesest tunnist viimaseeni. Kõigil tiimidel on ühine baasülesanne - väike veebirakendus, mis salvestab ja kuvab andmeid (näiteks ülesannete loend, broneerimine või hääletus) - ning iga tiim lisab sellele ühe omavalitud lisafunktsiooni. Ühine baas teeb ristülevaatus ja tunnis võrdlemise võimalikuks, lisafunktsioon annab omanikutunde. Projekti saab siduda G2 uurimistöö või digilahendusega.





	Nõuded ei ole algusest peale lõplikult paigas: 5. tunnis lisandub ülesandele üks uus nõue, nii et tiimid peavad spetsifikatsiooni ja agendi tööülesannet uuendama - täpselt nagu päris projektis. Koodi ülevaatus käib kogu kursuse vältel: alates 2. tunnist töötavad tiimid harudes ja pull request'idega ning paarid vaatavad üksteise koodi Gitis üle ja kommenteerivad seda.
Kursuse tulemuse kujunemise viis	AI-ta lugemiskontrollid igal tunnil (väljundi ennustamine, vea leidmine). Arenduspäevik: jagatud AI-vestlused + 3 lauset refleksiooni iga tunni lõpus; õpetaja loeb juhuvalikuga. Ristülevaatus Gitis alates 2. tunnist: paar A vaatab üle paar B koodi lühikese kontroll-loendi alusel (loetavus, testid, kas sisend on kontrollitud) ja kommenteerib pull request'is. Hinnatakse ülevaatus panust, mitte ainult oma koodi. Minikaitsmised igal tunnil: 2 õpilast selgitavad ühe rea või otsuse. Praktilised tööd igal kohtumisel; testid kuuluvad töö juurde. Kursuse lõputöö: tiimiprojekti kaitsmine (demo, dokumentatsioon ja koodi selgitamise vestlus, kus õpetaja valib rea). Kaitsmised on jaotatud 9. ja 10. tunni vahel. Kursus on edukalt sooritatud, kui praktilised tööd on esitatud (vähemalt 51%), ristülevaatused tehtud





ning tiimiprojekt esitatud ja kaitstud (töötav prototüüp, testid, dokumentatsioon, esitus). Hindamisel on määrav see, mida AI õpilase eest teha ei saa: spetsifikatsiooni täpsus, testid, kaaslase koodi ülevaatus ja oma lahenduse suuline põhjendamine. Kood iseenesest ei ole hinde alus.





Õpitulemused	Kursuse läbinud õpilane: - kirjeldab tarkvaraarenduse elutsükli etappe ja oskab väikest projekti planeerida; - kirjeldab projekti ülesande nõuetena ja koostab spetsifikatsiooni, mille järgi saab töötada nii inimene kui ka AI-agent; - uuendab spetsifikatsiooni ja agendi ülesannet, kui nõuded töö käigus muutuvad; - struktureerib koodi klasside ja moodulite abil ning oskab nimetada erinevate keelte (TypeScript, Python, Golang) erinevusi; - ehitab lihtsa serveripoolse API ning selgitab, kuidas HTTP ja JSON töötavad; - kavandab lihtsa andmemudeli, salvestab andmeid püsivalt ja katab andmekihi testidega; - ühendab kasutajaliidese ja serveripoolse terviklikuks veebirakenduseks; - juhib AI-agenti arendustöös: annab ülesande, kontrollib tulemust, märkab agendi vigu ja parandab need; - teeb kaaslaste koodile ülevaatuset Gitis kontrollloendi alusel (loetavus, testid, sisendi kontroll) ja põhjendab oma kommentaare; - tunneb turvalisuse põhimõtteid ja oskab nimetada, kuidas agendi tööd päris tarkvarafirmades kontrollitakse; - avaldab rakenduse internetis ja kasutab Giti tiimitöös (harud, pull request); - töötab tiimis, dokumenteerib lahendust ja kaitseb seda esitlusel, selgitades suvalist rida koodist.
Vajalikud õppematerjalid ja -vahendid	Visual Studio Code, Node.js, TypeScript Kodeerimisagent (nt Codex) ja vestlusmudel (nt ChatGPT) Git, GitHub: harud, pull request, koodi ülevaatus ja kommenteerimine, automaatsed kontrollid Rakenduse avaldamise keskkond (nt GitHub Pages / pilveteenus) Ristülevaatuset kontrollloend (loetavus, testid, sisendi kontroll) Arenduspäevik (jagatud AI-vestlused + refleksioon) Google Classroomis



	Google Classroom, õpetaja materjalid ja esitlused Arvutiklass või oma sülearvuti
--	---

4.

Kursuse ülesehitus:

Teema	Tundide arv	Märkused
1. Projekti algus: ülesanne ja nõuded Praktiline töö: projekti kirjeldus ja spetsifikatsioon v1	2 x 70 min	Kursuse eesmärgid, õpitulemused, hindamiskriteeriumid ja õppekorraldus. Tiimid (2 õpilast) saavad ühise baasülesande ja valivad sellele oma lisafunktsiooni. Tiim kirjutab ülesande lahti nõueteks ja koostab spetsifikatsiooni v1. Täpne spetsifikatsioon on agendi juhtimise tähtsaim tööriist - siin see selgub.
2. Koodi struktuur ja tiimitöö Gitis Klassid ja moodulid; harud ja pull request'id Praktiline töö: algstruktuur + esimene ristülevaatus	2 x 70 min	Miks suur kood tükkideks jagatakse. Kõrvalpilk Pythonile ja Go-le. Agent teeb algstruktuuri spetsifikatsiooni järgi; tiim selgitab, miks just nii. Ristülevaatus algab kohe: paar A vaatab paar B koodi Gitis üle ja kommenteerib. Nii käib see kogu kursuse.
3. Server ja API HTTP ja JSON Praktiline töö: agent ehitab, tiim kontrollib iga otspunkti käsitsi	2 x 70 min	Mis tegelikult juhtub, kui veebileht laeb? Agendi tööd ei võeta usu peale: iga otspunkt kutsutakse käsitsi välja ja vastus kontrollitakse.



4. Andmete salvestamine ja andmemudel Praktiline töö: testid andmekihile	2 x 70 min	Lihtne andmemudel ja andmete püsiv hoidmine. Mis juhtub, kui andmed on valed või puudu?
5. Terviklik rakendus v1 ja nõuete muutus	2 x 70 min	Kasutajaliides + server + andmed koos: esimene töötav versioon. Ülesandele lisandub üks uus nõue. Tiim uuendab spetsifikatsiooni ja annab agendile muudetud tööülesande - nõuded muutuvad ka päris projektis töö käigus.
6. Kui agent eksib Praktiline töö: tahtlikud ebaõnnestumised	2 x 70 min	Agent saab ebamäärase ülesande, vale konteksti, vastuolulised nõuded. Mis juhtub? Kuidas märkad? Kuidas parandad? Lahkame, mida agent päriselt tegi.
7. Kvaliteet ja turvalisus Põhjalik ristülevaatus kontroll-loendi alusel	2 x 70 min	Kontroll-loend: loetavus, testid, kas sisend on kontrollitud, mis juhtub vigase sisendiga. Turvalisuse alused; kuidas agendi tööd kontrollitakse päris tarkvarafirmades. Leitud probleemid lähevad parandusse ja parandus omakorda ülevaatusse.
8. Avalikustamine ja automaatika	2 x 70 min	Ühe nupuga deploy - ainult nii palju DevOps, et rakendus töötab internetis ja seda saab päriselt kasutada. Automaatsed kontrollid: testid jooksevad iga pull request'i peal.
9. Arendussprint ja esimesed kaitsmised	2 x 70 min	Tiim juhib agent'i nagu nooremarendajat: ülesanded, ülevaatus, testid. Tunni teine pool: pool tiimidest kaitseb (8 min tiimi kohta).
10. Kaitsmised ja kokkuvõte	2 x 70 min	Ülejäänud tiimid kaitsevad: demo, dokumentatsioon ja koodi selgitamise vestlus, kus õpetaja valib rea. Kokkuvõte ja tagasiside; arutelu, kelle töö see 2030. aastal on.
Kokku	20 x 70 min (10 kohtumist)	



